Using Proof-Planning to Investigate the Structure of Proof in Non-Standard Analysis

Ewen Maclean

Doctor of Philosophy

Centre for Intelligent Systems and their Applications
School of Informatics
University of Edinburgh

2004

Abstract

This thesis presents an investigation into the structure of proof in non-standard analysis using proof-planning. The theory of non-standard analysis, developed by Robinson in the 1960s, offers a more algebraic way of looking at proof in analysis. Proof-planning is a technique for reasoning about proof at the meta-level. In this thesis, we use it to encapsulate the patterns of reasoning that occur in non-standard analysis proofs.

We first introduce in detail the mathematical theory and the proof-planning architecture. We then present our research methodology, describe the formal framework, which includes an axiomatisation, and develop suitable evaluation criteria. We then present our development of proof-plans for theorems involving limits, continuity and differentiation. We then explain how proof-planning applies to theorems which combine induction and non-standard analysis.

Finally we give a detailed evaluation of the results obtained by combining the two attractive approaches of proof-planning and non-standard analysis, and draw conclusions from the work.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Ewen Maclean*)

Acknowledgements

I am enormously indebted to my supervisors Jacques Fleuriot and Alan Smaill for their unfaltering support and invaluable guidance. I am also hugely grateful to all of the members of the DReaM group, past and present, who have provided me with such an enjoyable and stimulating research environment, and great friendship; to all of the members of Felicidade and the Latin Quarter whose talents have brought my music to life; to my great friend and colleague Ruli, who started and ended his thesis on the same day as me; and finally to my family who have somehow always managed to put up with me.

This work was funded by the EPSRC, award number 99303126.

Table of Contents

List of Figures

Chapter 1

Introduction

The mechanisation of mathematical proof is a field of research which has gained success in many areas such as verification, and for many different mathematical theories such as induction. One of the main challenges of mechanisation is the automation of such proof. We present here a study into the feasibility of automating proof from the mathematical domain of *non-standard analysis*. We are particularly interested in finding common structure in non-standard analysis, and use the λ*Clam proof-planner* to investigate the nature of such structure. The attraction of combining proof-planning and non-standard analysis is that they both simplify the problem of automating real analysis proofs. Non-standard analysis formalises the idea of an infinitesimal, used informally by Newton and Leibniz, and proof-planning allows us to encapsulate this informal reasoning.

1.1 Background to the idea

Proof-planning is a technique for automating proof which has been successfully applied to a large corpus of theorems from induction, and more recently to problems from First Order Temporal Logic [Castellini and Smaill, 2001], finding loop invariants in imperative programs [Stark and Ireland, 1998] and in [Janičić et al., 1999, Janičić and Bundy, 2002] for the automatic synthesis of decision procedures. In this thesis we exploit the expressive powers of the proof-planning to investigate the structure of proof in a new mathematical theory– non-standard analysis.

Following the explicit formalisation of the Ultrapower Construction for non-standard analysis [Robinson, 1966] performed in Isabelle/HOL by [Fleuriot, 2001a], the proofs of many

standard analysis theorems were mechanised using non-standard analysis. Many of the proofs in this work benefitted from the automatic tactics in Isabelle, but there remains the challenge of automating the harder parts of the proofs. The aim of the thesis is then to use proof-planning to encapsulate the common patterns of reasoning that occur in these more difficult parts of the proofs.

1.1.1 Mechanisation of mathematical theory

The initial task of the thesis is to specify a formal framework by which to reason about non-standard analysis. Our approach is to take the formal construction of [Fleuriot, 2001a] and to produce an axiomatisation based on the higher level theorems proved in that work. We construct such an axiomatisation, presented in chapter 4, on which we justify the soundness of the proof-plans we yield.

1.1.2 Automation using Proof-planning

Proof-planning allows us to reason at a more abstract level than with an object-level theorem prover. We can specify *methods* which transform a goal in a way which corresponds to the application of many tactics. We can also specify *critics*, which analyse the failure of a method and suggest some course of action to allow the proof-plan to proceed. The way in which we combine methods and critics is by using a *proof-plan*.

In our work we develop methods, critics and proof-plans to account for a number of real analysis theorems. We are interested in encapsulating the structure of proofs using non-standard analysis, and the methods and critics we develop account for large parts of the associated object-level proof. We do not attach an object level prover, and so we cannot claim that we yield proof-plans which correspond to a sequence of tactic applications. However, we present an axiomatisation for our work, and show at each point which axioms are being used, allowing us to ascertain that it is possible to yield object-level proofs from our proof-plans.

1.2 Specific goals

The aim of this work is to advocate the use of proof-planning as a tool for automating proof by accounting for common proof structure, and also to show that non-standard analysis proofs of real analysis theorems contain such structure, and are hence a good candidate for automation.

1.2.1 Limit Theorems

The first family of proofs we investigate includes high level theorems about limits and continuity. We describe how non-standard analysis provides a simpler characterisation of limit and continuity and exploit this in the design of the proof-planning machinery. Using the $\lambda Clam$ proof-planner allows us to reason naturally about higher order concepts, and we make use of this higher order capability by studying theorems which involve arbitrary functions. For example we study such theorems as *LIM* $\times$, which states that the product of the limits of two functions at a point is equal to the limit of the product of the functions, characterised by

$$\lim_{x \to a} f(x) = l_f \wedge \lim_{x \to a} g(x) = l_g \vdash \lim_{x \to a} f(x) \times g(x) = l_f \times l_g.$$

Here we are reasoning about arbitrary functions, and we can see from the work presented in chapter 5 that when finding a proof-plan for this theorem using non-standard analysis, we exploit the higher order functionality of $\lambda Clam$.

1.2.2 Theorems involving induction

We introduce a different approach for finding proof-plans for a number of theorems in real analysis which are existentially quantified. We follow techniques from computable analysis, such as those presented in [Bishop and Bridges, 1985], and use non-standard analysis to simplify some of the reasoning. The central notion in this work is one of *partitioning*, which we explain and describe in more detail in chapter 6.

1.3 Research contribution

We outline three major areas where we have made a research contribution through this work. It is important to mention here that proof-planning does not yield object-level proofs, but instead gives a description of how the proof should proceed at a more abstract level.

1.3.1 Implementation

We use the proof-planner $\lambda Clam$, in which we have provided an axiomatisation for non-standard analysis, and a set of methods and critics by which to reason about proofs. In the course of this work, we have also implemented a number of proof-planning devices which are now part of the main $\lambda Clam$ system. These are described in detail in chapters 5 and 6, when we cover the specific families of proofs we study.

1.3.2 New and readable proofs

We present what we believe to be a new proof of a slightly modified version of Rolle's Theorem, and construct a proof-plan for it in $\lambda Clam$. Some of the theorems for which we construct proof-plans have been proved in other settings. In particular [Bledsoe and Ballantyne, 1977] had an automated theorem prover based on resolution which was capable of proving many of the theorems we present in chapter 5. We argue that due to the resolution style, their proofs are not very easy to read, and do not follow the sorts of patterns of reasoning that a human would perform. Our work, by contrast, produces proof-plans which correspond more closely to the strategies a human might adopt during proof.

It must be noted here however, that the proof-plans we produce are not executed in an object-level theorem prover, and so a direct comparison between our work and that of Bledsoe and Ballantyne is not possible.

1.3.3 Understanding of the structure of proofs

The work presented in this thesis provides a fundamental understanding of the structure of certain analysis proofs using non-standard concepts. Using proof-planning we are able to study the general proof techniques which can be used to tackle analysis theorems using non-standard concepts. Investigating proofs of the theorems we present in this thesis has allowed us to develop general purpose techniques for automation. Since the issue of automation is contentious in this work due to the lack of explicit object level proofs we must provide an argument for the soundness of the proof-plans yielded. We discuss the validity of our approach further in section 7.3.

1.4 Organisation of the thesis

In the next chapter we describe the mathematical background to our work. We highlight the various attempts at finding a formal basis for the sort of informal reasoning performed by Newton and Leibniz, with particular reference to the differene between formal constructions of number systems such as the real numbers, and axiomatic approaches. Chapter 3 presents an overview of the literature that exists on theorem proving in analysis. We present interactive and automatic work done on both standard and non-standard analysis.

In chapter 4 we describe our methodology, and introduce our logic and axiomatisation. We introduce first our research hypothesis and research goals, and describe some nomenclature for

the proof-planning entities we describe. We then go on to introduce our type system, and our axiomatisation for all of the types we introduce. We discuss some theoretical issues, and give a detailed description of our evaluation scheme.

Chapters 5 and 6 rely heavily on the both the axiomatisation and the evaluation scheme set out in chapter 4. Chapter 5 describes the theorems involving limits and continuity which we study. We introduce representations of the proof-plans developed for each theorem, and then describe the methods and critics we develop in order to construct a proof-plan which accounts for as much of the automation as possible. We perform some experiments with more naïve search strategies, and give an evaluation of the results, comparing the work with other systems.

Chapter 6 introduces induction and a different technique for constructing proof-plans for analysis. We describe the technique using Rolle's Theorem as an example. We give descriptions of the proof-plans yielded for each part of the theorem, and present the proof architecture which we capture using proof-planning. We describe the methods and critics we develop, and show how we construct proof-plans to account for the theorems.

In chapter 7 we draw some conclusions about the work, and describe some further avenues of research which have not been fully investigated in this thesis.

Finally in the appendices we include some sample code and proof output from λ*Clam*, together with a description of the proof-plan yielded for one of the important theorems described in chapter 6.

Chapter 2

Analysis

In this chapter we review the background of the central mathematical ideas of this thesis. We start by motivating the construction of non-standard analysis through a brief description of the history behind the notion of the "infinitesimal" and describe the most important concepts involved. We then introduce the real numbers, paying particular attention to the difference between axiomatic approaches, and formal constructions. We go on to describe the important concepts involved in any theory of the reals. Finally we describe three different versions of non-standard analysis which provide a formal basis for introducing the notion of the "infinitesimal". We discuss some of the properties that these theories have, in particular with relation to a more algebraic notion of limit.

2.1 Brief Historical background for NSA

In the eighteenth century, Newton and Leibniz both relied on a concept of an infinitesimal in order to reason about derivatives. The problem with this work was that the number system they used did not contain the infinitesimal quantities which their proofs relied on, and as such did not have a sound mathematical basis. The issue was so philosophically controversial that Bishop Berkeley wrote an attack on the use of infinitesimals in [Berkeley, 1734], ending his attack with a series of questions, the 54th of which being

> ...whether the same things which are now done by infinities may not be done by finite quantities? And whether this would not be a great relief to the imaginings and understandings of mathematical men?

Leibniz tried to solve this problem by constructing a number system which included both infinite and infinitesimal quantities, using as inspiration the construction of the imaginary num-

bers. The problem with inventing a number system which includes infinitesimal numbers is that some formal way of stating how to define infinitesimals is needed. When this failed to materialise, interest in infinitesimal reasoning waned, and a different approach was taken up by mathematicians like Cauchy and Weierstraß.

Cauchy gave the first rigorous development of mathematical analysis, basing his work on the notion of a limit– a process, viewed geometrically, where a point was approached so that it always got nearer, but was never reached. This notion, which had been used by Newton and d'Alembert although not rigorously, satisfied the doubters of infinitesimal reasoning. Weierstraß developed a formal notion of limit, known as the ε-δ formulation, which has become the standard analysis teaching method.

The issue of infinitesimals was left dormant for some time until Robinson applied some well known results from mathematical logic to the problem of constructing a number system which included infinite and infinitesimal quantities. Robinson used a set theoretic concept known as an *ultraproduct* to construct such a number system. This was the number system that Leibniz had been looking for to justify his work. This application of model theoretic results gave rise to non-standard analysis.

2.2 Standard Analysis

Standard analysis studies the properties of functions over the real numbers. We present here some of the choices there are in formulating a real number system, and outline a brief description of a formal construction of the real numbers using just set theory. We do this here to provide context for our presentation of non-standard analysis, shown in section 2.3.1.

2.2.1 Properties of reals

We want to be able to make use of a number system which has various important properties. Firstly we want the real numbers to form a real closed field. The field axioms, which underlie the main properties we need for the real numbers, include group axioms about multiplication and addition, and rules defining an order on the numbers. Given just the ordered field axioms, there is nothing to distinguish the real numbers from the rational numbers. A full description of the axioms for a *real closed field* is given in section 4.2.4, where we present the axiomatisation we adopt for this work. The important properties which define the number system to be the real numbers are

The completeness of the reals

This states that any non-empty set of real numbers which has an upper bound, has a least upper bound. This is not true for the rationals, evident in the fact that $\sqrt{2}$ is not a rational number. The *supremum* property which characterised the completeness is given in higher order logic as follows. For any property P of the reals,

$$\begin{aligned}(\exists x \in \mathbb{R}.\ P(x)) \wedge (\exists U \in \mathbb{R}.\ \forall y \in \mathbb{R}.\ P(y) \rightarrow y \leq U) \rightarrow \\ \exists u \in \mathbb{R}.\ (\forall x \in \mathbb{R}.\ P(x) \rightarrow x \leq u) \wedge \\ \forall u' \in \mathbb{R}.\ (\forall x \in \mathbb{R}.\ P(x) \rightarrow x \leq u') \rightarrow u \leq u'\end{aligned}$$

The Archimedean property of the reals

This states that every real number is bounded above by a natural

$$\forall x \in \mathbb{R}.\ \exists n \in \mathbb{N}.\ \ x < n.$$

The consequences of this property are that the real numbers cannot contain infinitesimal quantities. If there were such a number, then its inverse would be greater than any natural number, which cannot be the case.

There are other noteworthy axiomatisations of the reals, which differ from the standard real closed field axiomatisation presented in section 4.2.4. In particular, there are constructive axiomatisations of the real numbers. These are of particular interest because using a constructive real number system allows algorithms to be extracted from proofs. The axiomatisation for the constructive real numbers is in general characterised by adding an extra concept of *nearness*, as in the works of [Beeson, 1985, Cruz-Filipe, 2002], or *apartness* as in [Troelstra, 1973] for example.

2.2.2 The construction of the real numbers

It is often of interest to mathematicians to verify that a mathematical theory can be constructed purely from the axioms of a set theory. We give a brief presentation here of how this is possible, in order to motivate the choice of representation chosen later for non-standard analysis.

Given a definition of the natural numbers, $\mathbb{N}$, we can construct the integers, the rationals and the real numbers. There is more than one way of doing this, but we consider here just a method motivated by Cauchy sequences, of considering equivalence classes of converging sequences of rationals.

We define an equivalence relation $\sim$ on pairs of naturals to be

$$< x_1, y_1 > \sim < x_2, y_2 > \iff x_1 + y_2 = x_2 + y_1.$$

The integers $\mathbb{Z}$ can then be defined by the equivalence class of all pairs $< x, y >$ under $\sim$. We can show that the definitions for addition and multiplication are closed for this equivalence class and are hence defined for $\mathbb{Z}$.

We can further define an equivalence relation $\sim'$ on pairs of integers, whose second element is non-zero, to be

$$< x_1, y_1 > \sim' < x_2, y_2 > x_1 \times y_2 = x_2 \times y_1$$

The rationals $\mathbb{Q}$ can then be defined by the equivalence class of all pairs $< x, y >, y \neq 0$, under $\sim'$. This can be shown to be well behaved for addition and multiplication also. The rationals can be shown to be dense i.e. that between any two rationals there exists another rational. The famous proof that $\sqrt{2}$ is irrational shows that there is no number which is a solution of the polynomial $x^2 - 2 = 0$. What is possible is to construct an infinite sequence of rationals that approaches such a number arbitrarily closely. In order to do this, we define a *Cauchy sequence*

Definition 1 *An infinite sequence over the rationals is a Cauchy-sequence, if its members approach each other arbitrarily closely. Formally, $f : \mathbb{N} \to \mathbb{Q}$ is a Cauchy sequence if and only if*

$$\forall \varepsilon \in \mathbb{Q}.\ \varepsilon > 0 \to\ \exists N \in \mathbb{N}.\ \forall m, n \in \mathbb{N}.\ m \geq N \wedge n \geq N\ \to\ |f(m) - f(n)| < \varepsilon$$

Stated simply, for any positive rational ε, there is some natural N such that for any point in the sequence after N, the difference between subsequent elements of the sequence is less than ε. The idea is to define the set of all Cauchy sequences of rationals, and to define an equivalence relation between sequences, which is true for sequences that approach each other arbitrarily closely. Each such equivalence class denotes a real number with arithmetic being defined pointwise for each rational number in the sequence. Now the real number we yield is the limiting position of the ε width "tunnel".

2.2.3 Intuitive proofs

At school we are often taught to write the following fraction to find the derivative of a function f at a point x:

$$\frac{\delta f(x)}{\delta x} = \frac{f(x + \delta x) - f(x)}{\delta x}$$

and then told to consider what happens when δx approaches 0. The notion of dx, some fictional quantity representing an infinitesimal is then introduced to write the derivative of f at the point x to be

$$\frac{df(x)}{dx} = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$$

where a limit is a loosely defined concept meaning that h is considered when it is very small. This characterisation of differentiation becomes natural, and people understand the notion of limit intuitively, and can reason about behaviours in very small neighbourhoods of points. The problem is that using infinitesimals to help reason in very small neighbourhoods is ill-defined. The real number system does not contain such quantities, so we need some system for formalising the notion of "infinitesimal".

2.2.4 Notions of limit and convergence

In the nineteenth century Karl Weierstraß developed the theory of standard analysis by formalising the notion of the "neighbourhood" of a point. He first devised the ε-δ formalisation of limits in standard analysis. Stated formally we have the following definition: The limit l, of a function $f : \mathbb{R} \to \mathbb{R}$ at a point a is defined as

Definition 2

$$(\lim_{x \to a} f(x) = l) \equiv$$
$$(\forall \varepsilon \in \mathbb{R}.\ \varepsilon > 0 \to \exists \delta \in \mathbb{R}.\ \delta > 0 \wedge \forall x \in \mathbb{R}. \quad 0 < |x - a| < \delta \to |f(x) - l| < \varepsilon)$$

and is the crux of most standard analysis proofs. This is the main idea used in most analysis proofs because it formalises the notion of a function approaching a certain value at a certain point. It is not immediately clear why this definition should demonstrate this. The best way to view it is by considering a picture of a unary function approaching a value at a certain point a as in figure 2.1. The limit definition can be interpreted graphically by considering the quantities ε and δ on figure 2.1. The definition states that for every length ε, a length δ can be chosen such that for all points from $a - \delta$ to $a + \delta$, the difference between the limit l and the function value will be less than ε. The idea of continuity of functions follows easily from the definition of a limit.

Definition 3 *A function $f : \mathbb{R} \to \mathbb{R}$ is continuous at x_0 if*

$$\lim_{x \to x_0} f(x) = f(x_0)$$

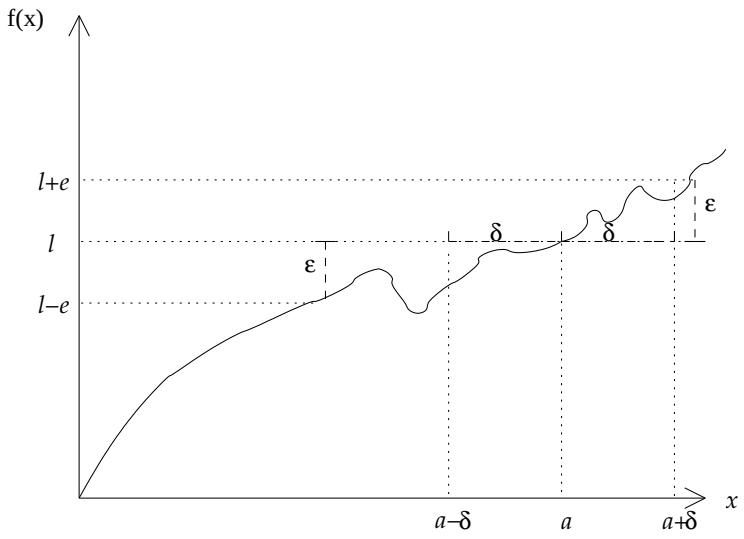

Figure 2.1: A function with limit l at $x = a$

We also yield the following definition:

Definition 4 *A function* $f : \mathbb{R} \to \mathbb{R}$ *is uniformly continuous if*

$$\forall \varepsilon \in \mathbb{R}.\ \varepsilon > 0 \to \exists \delta \in \mathbb{R}.\ \delta > 0\ \wedge \forall x, y \in \mathbb{R}\ \ |x-y| < \delta \to |f(x) - f(y)| < \varepsilon.$$

Now that limits have been formally defined, the notion of a derivative can be taken straight from Newton and Leibniz's work.

Definition 5 *The derivative* $f'(x)$ *of a function* $f : \mathbb{R} \to \mathbb{R}$ *at x is given by the limit expression*

$$f'(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$$

where $f'(x)$ can be interpreted as the function of the gradient of f at the point x. There are many concepts which make use of the notion of a limit, such as the definition of an integral.

As an example of a simple proof using standard analysis, and as a motivation for what follows, consider a proof of uniform continuity using ε-δ notation. Let us say that we want to prove that x^2 is uniformly continuous on the interval $[0,1]$ [1]. We can write

$$\begin{aligned} |f(x) - f(y)| &= |x^2 - y^2| \\ &= |(x-y)(x+y)| \end{aligned}$$

[1] Uniformly continuous on an interval means that the definition for uniform continuity holds for all points within the interval

$$\begin{aligned} &= |x-y||x+y| \\ &\leq |x-y|(|x|+|y|) \\ &\leq 2|x-y|. \end{aligned}$$

We are now faced with the task of finding a δ such that for any value of ε

$$|x-y| < \delta \rightarrow |f(x)-f(y)| < \varepsilon.$$

Clearly if we choose $\delta = \frac{\varepsilon}{2}$ this will be satisfied, because $|f(x)-f(y)| \leq 2|x-y|$.

The reasoning involved in this proof is typical of many proofs in standard analysis. The common formulation of limits and continuity in standard analysis, means that it is always the case that a δ has to be found which can satisfy the inequality involving the ε term. In this case a δ was found which was a function of the ε term. In proofs which demonstrate more general results, often a δ can be shown to exist which satisfies the inequality involving the ε term, but where the specific instantiation is not found.

2.3 Versions of non-standard analysis

We first present in this section the construction of the hyperreal number system which Robinson developed. This follows some of the same ideas as that of the development of the reals, and we present this first to show the rigorous formal basis for the hyperreals, and as a result for the theory of non-standard analysis. As with the real numbers, there is more than one approach to introducing the hyperreals. Here we present overviews of four such approaches: an extensional formal construction using set theory (i.e. one which enumerates the members of the set); an intensional set theoretic approach (i.e. one which refers to a defining property of the set); constructive version of an axiomatisation for non-standard analysis; and finally one motivated by the introduction of a nilsquare infinitesimal. We end the chapter by presenting definitions for limit and continuity which can be shown to be equivalent to standard definitions shown in section 2.2.4.

2.3.1 Ultrapower construction

The construction of the real numbers can be formalised by considering converging sequences of rationals. Hyperreal numbers can likewise be obtained by considering sets of sequences of real numbers, much in the same way that real numbers can be obtained by considering sets of sequences of rationals. The method outlined here is called the ultrapower construction.

Definition 6 *An* ultrafilter $\mathcal{F}$ *on* I *is a nonempty collection of subsets of* I, $\mathcal{F} \subseteq \mathcal{P}(I)$, *where* $\mathcal{P}(I)$ *is the set of all subsets of* I*:*

$$\mathcal{P}(I) = \{\mathcal{A} : \mathcal{A} \subseteq I\}$$

An ultrafilter $\mathcal{F}$ *must satisfy the following properties:*

1. *if* $A, B \in \mathcal{F}$, *then* $A \cap B \in \mathcal{F}$
2. *if* $A \in \mathcal{F}$ *and* $A \subseteq B \subseteq I$, *then* $B \in \mathcal{F}$
3. $\emptyset \notin \mathcal{F}$
4. *for any* $A \subseteq I$, *either* $A \in \mathcal{F}$ *or* $I - A \in \mathcal{F}$

A *free*, or *nonprincipal* ultrafilter is one which does not contain any finite sets. The Ultrafilter Theorem guarantees the existence of a free ultrafilter where I is infinite [Robinson, 1996]. The set of hyperreals can be defined by considering equivalence classes of sequences with respect to a free ultrafilter on the natural numbers ($\mathbb{N}$) defining the ordering of these sequences. A hyperreal number is represented by an equivalence class of sequences of real numbers $< r_n >$, and equivalence is defined with respect to the chosen nonprincipal ultrafilter on $\mathbb{N}$. The type of equivalence on sequences is called the "Almost-Everywhere Agreement" and is defined as

$$< r_n > \equiv < s_n > \iff \{n \in \mathbb{N} : r_n = s_n\} \in \mathcal{F}.$$

So two sequences of reals are equivalent if their indexing set i.e. the set of natural numbers containing the positions at which they agree, is in the ultrafilter $\mathcal{F}$. Because of the axioms defining an ultrafilter, this equivalence is unique modulo the choice of ultrafilter. To show that this is a well-defined representation of the hyperreals, one must show that it contains the reals, and that it is an ordered field. The set of equivalence classes (the *quotient* set), of $\mathbb{R}^{\mathbb{N}}$ is

$${}^*\mathbb{R} = \{[r] : r \in \mathbb{R}^{\mathbb{N}}\}$$

where $[r]$ is the equivalence class of sequence $r \in \mathbb{R}^{\mathbb{N}}$:

$$[r] = \{s \in \mathbb{R}^{\mathbb{N}} : r \equiv s\}.$$

Now to show that ${}^*\mathbb{R}$ is an ordered field one needs to give well-defined definitions for addition and multiplication of sequences, and a well-defined definition for the order relation $<$:

$$[r] + [s] = [< r_n + s_n >]$$
$$[r] \times [s] = [< r_n \times s_n >]$$
$$[r] < [s] \iff \{n \in \mathbb{N} : r_n < s_n\} \in \mathcal{F}.$$

We introduce here another piece of useful notation:

Definition 7

$$\| < r_n > \sim < s_n > \| \equiv \{n \in \mathbb{N} : r_n \sim s_n\}$$

where $\sim$ *is some relation on* $\mathbb{R} \times \mathbb{R}$ *and* ${}^*\mathbb{R} \times {}^*\mathbb{R}$. $\| < r_n > = < s_n > \|$ *gives the indexing set of the sequences* $< r_n >$ *and* $< s_n >$.

With these definitions, it is relatively easy to show that ${}^*\mathbb{R}$ is an ordered field with additive identity $[0]$ and multiplicative identity $[1]$. Now it remains to show that there is an order preserving injective map $\widehat{\ }: \mathbb{R} \to {}^*\mathbb{R}$. To illustrate this one can represent a real number $r \in \mathbb{R}$ in the hyperreals ${}^*\mathbb{R}$ as the constant sequence $\widehat{r} = [r] = [< r, r, ... >]$. For $r, s \in \mathbb{R}$:

$$\widehat{r+s} = \widehat{r} + \widehat{s}$$
$$\widehat{r \times s} = \widehat{r} \times \widehat{s}$$
$$\widehat{r} < \widehat{s} \iff r < s$$
$$\widehat{r} = \widehat{s} \iff r = s.$$

The *non-standard extension* of a function f defined over the reals, is the function ${}^*f$ which is extended to accept hyperreal arguments. This can be defined by

$${}^*f([< r_1, r_2, ... >]) = [< f(r_1), f(r_2), ... >]$$

Now infinite and infinitesimal numbers can be represented in the hyperreals. The interesting points about the definition of the ultrafilter are that either a set is in the ultrafilter or its complement is, and that the empty set is not contained in it. This means that if $\mathcal{F}$ is an ultrafilter and $\{A_1, ..., A_n\}$ is a finite collection of disjoint sets and $A_1 \cup ... \cup A_n \in \mathcal{F}$, then exactly one of $A_i \in \mathcal{F}$. Because the ultrafilter chosen cannot contain finite sets, it must contain all cofinite sets from property 4 of the definition. Hence one can construct both infinite and infinitesimal numbers by considering the hyperreal $\varepsilon = < 1/(n+1) : n \in \mathbb{N} >$:

$$\|0 < \varepsilon\| = \{n \in \mathbb{N} : 0 < 1/(n+1)\} = \mathbb{N}.$$

As $\mathbb{N} \in \mathcal{F}$, $[0] < [\varepsilon]$ in ${}^*\mathbb{R}$. Now consider the set

$$\|\varepsilon < \widehat{r}\| = \{n \in \mathbb{N} : 1/(n+1) < r\}$$

for any positive $r \in \mathbb{R}$. This is clearly cofinite, as ε tends to 0 in $\mathbb{R}$. Hence this set is in the ultrafilter, and so $[\varepsilon] < [r]$ in ${}^*\mathbb{R}$. This shows that $[\varepsilon]$ is a positive infinitesimal. A similar

argument can be given for $[\omega] = [< 1, 2, 3... >]$ to show that infinite numbers can be represented in the hyperreals.

The non-standard natural numbers, or hypernaturals, can be defined in a similar way to the hyperreals. The hypernaturals can be defined by taking equivalence classes of sequences of natural numbers. The finite hypernaturals are just the natural numbers, but there are also infinite hypernaturals. In chapter 6 we extend functions which map from the natural numbers to the reals. We do this by defining the functional extension to include the map

$$^*f([< n_1, n_2, ... >]) = [< f(n_1), f(n_2), ... >].$$

2.3.2 Internal Set Theory

An axiomatic version of non-standard analysis has also been devised by Nelson, and is known as *Internal Set Theory* [Nelson, 1977]. The idea behind this version of non-standard analysis is to work in first-order logic and augment the axioms of Zermelo-Fränkel set theory by introducing a new predicate called **standard**, and its associated axioms. One can then mimic the formal constructions of the hyperreals by attributing sets to the types of object that appear in the non-standard model for analysis.

Nelson introduced the new predicate **standard** purely syntactically, and defines semantic classes of objects according to how this predicate appears. Every object of classical mathematics is now either standard or non-standard. Also one can define objects which are *external* to classical mathematics, as those which use the standard predicate. As an example, the set of *standard* naturals is an external entity because it cannot be defined without reference to the **standard** predicate.

Nelson augmented the axioms of Zermelo-Fränkel set theory with three axioms called *idealisation*, *standardisation* and *transfer*.

Idealisation If E is an internal object which is defined from standard objects, then E is standard.

Standardisation All elements of an internal set are standard if and only if this set is finite. Intuitively, if a set is infinite then it must contain non-standard elements, and if a set contains non-standard elements (introduced by the new predicate **standard**) then there is no restriction on the number of non-standard elements introduced, so the set must be infinite.

Transfer principle Let $P(x)$ be an internal expression relative to x. $P(x)$ is true for all x, if and only if $P(x)$ is true for all standard x.

These three axioms along with the axioms from Zermelo-Fränkel set theory allow us to construct a theory of non-standard analysis in which integers are said to be *limited* or *unlimited*. For any real number x, we have

- x is *limited* $\iff$ there is a standard integer greater than x,
- x is *unlimited* $\iff$ it is greater than any limited integer,
- x is *infinitesimal* $\iff$ $|x| < \frac{1}{n}$, for any limited integer n,
- x is *infinitely close* to y $\iff$ $x - y$ is infinitesimal.

Now one can show that any limited real is infinitely close to a standard real number, and hence introduce a *standard part* operator which returns the unique standard real infinitely close to any limited real number. This axiomatisation gives a calculus for non-standard analysis.

2.3.3 Constructive non-standard analysis

As is demonstrated in [Palmgren, 1995], it is possible to axiomatise a constructive version of non-standard analysis using the notion of sheaf-theoretic nonstandard models. This development of a model for non-standard analysis is similar to Nelson's Internal Set Theory described in section 2.3.2. Again in this setting we can define notions of transfer and idealisation, but instead of standardisation, the notion of *underspill* is introduced. This states that if a statement is true in the non-standard model for all non-standard elements, then there must be at least one standard element for which it is true.

The precise details of such an axiomatisation are too complex to describe here in detail. The differences between this approach and a non-constructive approach is that we interpret formulas using sheaf-theoretic semantics, as opposed to the usual Tarskian semantics, which means that the interpretation of certain logical symbols is more involved. This allows the logic for the model to be intuitionistic. Other papers which use this sheaf-theoretic approach include [Palmgren, 1997].

Other versions of constructive non-standard analysis include [Liu, 1980], which introduces a constant ∞ to the axioms of Zermelo-Fraenkel set theory. He shows that such a constant is not constructive in any number type, and indeed the Zermelo-Fraenkel set theory is classical, but shows how it is possible to extract constructive content from proofs in non-standard analysis.

2.3.4 Bell's infinitesimal calculus

Another axiomatic version of infinitesimal calculus has been formalised by Bell, which takes as its motivation the implicit use of nilsquare infinitesimals in physics [Bell, 1998]. In this work, Bell sets out the different philosophical and logical notions of a continuum and a point, and defines geometrically the features we expect from *smooth worlds*. Crucially he points out that the assumption of the law of the excluded middle is false in this formulation of the realm of continuous functions. He introduces the usual ring definitions of addition and multiplication, and assumes the existence of a square root for any positive real number. All of these rules hold with respect to the semantics of the geometric model he creates.

In the geometric model there are some statements which in a normal axiomatisation of the reals we expect to hold, but in this case do not. Precisely, the following conjecture is invalid:

$$\forall a, b.\ a \times b = 0 \ \rightarrow\ a = 0 \ \vee\ b = 0.$$

This is a result of not being able to assume the law of the excluded middle in proofs. He defines a set of infinitesimal quantities, Δ, and states the following principle:

Principle of Microaffineness For any map $g : \Delta \rightarrow \mathbb{R}$, there exists a unique $b \in \mathbb{R}$, such that $\forall \varepsilon \in \Delta$ we have:

$$g(\varepsilon) = g(0) + b \times \varepsilon$$

which states that g is a straight line with slope b.

What is important about this definition is that it constitutes purely *location* and *direction*. Δ consists of entities which are too short to constitute a line, but are nonetheless not a point. Bell identifies this with a "rigid rod" which can be rotated to be tangential to any part of a curve, but cannot be bent around the curve. He then goes on to prove the following important properties of the domain Δ:

1. Δ is included in the closed interval $[0,0]$, but is not identical with $\{0\}$.
2. Every element of Δ is indistinguishable from 0 i.e. cannot be determined to be to the left of right of 0.
3. It is *false* that for all $\varepsilon \in \Delta$, either $\varepsilon = 0$ or $\varepsilon \neq 0$.
4. Δ satisfies: $\forall a, b \in \mathbb{R},\ \varepsilon \in \Delta.$ if $\varepsilon \times a = \varepsilon \times b$, then $a = b$. In particular if $\forall \varepsilon \in \Delta.\ \varepsilon \times a = 0$ then $a = 0$.

Now we have a calculus where we have formally defined a *nilsquare* infinitesimal, i.e. $\exists \varepsilon \in \Delta$ such that $\varepsilon^2 = 0$, yet $\varepsilon \neq 0$. This follows from the four properties outlined above. This allows us to define derivatives in the normal way, and reason about "higher-order" infinitesimals in the way that physics proofs often do.

2.3.5 A brief comparison of the approaches

The approaches discussed in section vary in their formalisation and motivation. Bell's approach is motivated by physics problems and attempts to construct an axiomatisation by which proof can be performed using a notion of a nilsquare infinitesimal. The Ultrapower approach is a formal construction of the non-standard real numbers using the fundamental concepts of set theory. Internal Set Theory attempts to simplify the axiomatisation by using the intensional approach of adding a defining predicate to the axiomatisation by which a new number system can be defined.

2.4 Simplified definitions and proof

The reason why the ε-δ proofs are in general considered to be difficult is because of the need to express the notion of a limit, without appealing to infinitesimals. The notion of limit in standard analysis involves alternating quantifiers which make the proofs sometimes difficult to complete. We want to exploit the fact that such an infinitesimal element now exists in the hyperreal domain. We introduce some notions in non-standard analysis which will occur often elsewhere in this thesis.

In what follows we use macros for limits. *lim* is used to denote limits and *NSlim* is used to denote the macro for the non-standard characterisations of a limit. This is just a presentational macro and is not used internally in λ*Clam*. For this reason we do not use the macro notation when showing the proof-plans developed for the theorems in chapters 5 and 6.

2.4.1 Transfer Theorem

In the case of the ultrapower construction for the hyperreals, we can appeal to a very powerful result from logic, called Łoś's Theorem which is the key to using non-standard analysis to prove results from standard analysis [Hurd and Loeb, 1985]. We introduce the *transfer principle*, which serves both as a definition of the so-called *-transform and a theorem about its most important property.

Definition 8 *If ϕ is a first order statement expressed over the reals, then the *-transform of ϕ denoted ${}^*\phi$ is defined by applying the following rules:*

replace functions and predicates in ϕ by their non-standard extensions;

replace unquantified real numbers in ϕ by their embeddings in the hyperreals;

variables quantified over the reals in ϕ become quantified over the hyperreals in ${}^*\phi$.

Theorem 1 *Any statement ϕ expressed over the reals, $\mathbb{R}$, is true if and only if its transform ${}^*\phi$ is true over the hyperreals, ${}^*\mathbb{R}$.*

This principle allows results in standard analysis to be guaranteed to be true also in the non-standard domain. Also it means that if the transform of a theorem can be proved in the non-standard domain, then it is true in the standard domain.

2.4.2 Limits

The advantage of formally defining such an extension of the real numbers is that one can formally state what it means to be in an "infinitesimal neighbourhood". This is defined using the so-called "infinitely close" relation denoted by $\approx$. This relation simplifies definitions and proofs, as will be seen when we consider the non-standard versions of the definitions presented in section 2.2.4.

Definition 9 *The limit of a function $f : \mathbb{R} \to \mathbb{R}$, with limit l at a is given in non-standard analysis by*

$$(\text{NSlim}_a\ f = l) \quad \equiv (\forall x \in {}^*\mathbb{R}\ \ x \approx \widehat{a} \wedge x \neq \widehat{a} \to {}^*f(x) \approx \widehat{l}) \tag{2.1}$$

where $\widehat{a}$ and $\widehat{l}$ are the embeddings of the real numbers a and l respectively in the hyperreals, and ${}^*f$ is the non-standard extension of the function f.

This definition matches exactly our intuitive understanding of the more abstruse ε-δ formulation of a limit. Indeed, it simply says that in an infinitesimal neighbourhood, a continuous function will also take values within an infinitesimal neighbourhood. The corresponding theorem which is easily proved by the transfer principle is: (see section 4.3.2)

Theorem 2

$$\lim_{x \to a} f(x) = l \quad \Longleftrightarrow \quad (\text{NSlim}_a\ f = l).$$

2.4.3 Continuity

The definition of continuous at a point can be unfolded using the non-standard definition of a limit. Throughout this thesis however, we use uniform continuity as it has a very simple characterisation in non-standard analysis.

Theorem 3 *A function* $f : \mathbb{R} \to \mathbb{R}$ *is uniformly continuous if and only if*

$$\forall x, y \in {}^*\mathbb{R}.\ \ x \approx y \to {}^*f(x) \approx {}^*f(y).$$

We refer to this characterisation as uniform continuity since it can be defined at infinite x and y. This theorem provides a much more intuitive and simple understanding of uniform continuity.

Although we use uniform continuity in general, there is an analogous result which characterises continuity:

Theorem 4 *A function* $f : \mathbb{R} \to \mathbb{R}$ *is continuous if and only if*

$$\forall x, y \in {}^*\mathbb{R}.\ \ finite(x) \wedge x \approx y \ \ \to \ \ {}^*f(x) \approx {}^*f(y) \quad .$$

Continuity cannot be defined at infinite hyperreals.

2.4.4 Differentiability

In non-standard analysis one can use the non-standard definition of a limit to rewrite the standard definition of a derivative given in section 2.2.4. We can then state the theorem

Theorem 5 *The derivative of a function* $f : \mathbb{R} \to \mathbb{R}$ *at a point x is* $f'(x)$ *if and only if:*

$$\forall h \in {}^*\mathbb{R}.\ \ h \neq 0 \wedge h \approx 0 \to \frac{{}^*f(\widehat{x}+h) - f(\widehat{x})}{h} \approx f'(\widehat{x})$$

This characterisation of a derivative can be further generalised by considering *uniform differentiability*. When transferred this gives us a definition for the extended derivative function ${}^*f'$ evaluated at an arbitrary hyperreal point:

$$\forall x, h \in {}^*\mathbb{R}.\ \ h \approx 0 \wedge h \neq 0 \to \frac{{}^*f(x+h) - {}^*f(x)}{h} \approx {}^*f'(x).$$

We use this generalised notion when considering the real analysis theorems presented in chapter 6. Support for our approach is provided by [Hoskins, 1990], for example, where it is argued that uniform continuity and differentiability are more natural ways of reasoning about analysis proofs.

2.4.5 On the notion of "limit"

The majority of the preceeding definitions make use of the concept of limit, and so transfer directly into the non-standard realm from the standard domain. The proofs become much simpler in non-standard analysis fundamentally because we lift the restriction of the field being Archimedean, and hence we can introduce an infinitesimal element.

As an example of a simple proof in non-standard analysis, consider once again the problem of proving $f(x) = x^2$ to be uniformly continuous in the interval $[0,1]$. When stated over the hyperreals, this means that the non-standard extension of f has to be uniformly continuous over the hyperreals in the non-standard extension ${}^*[0,1]$ of the interval $[0,1]$.

$$\forall x, y \in^* \mathbb{R}.\ 0 \leq x \leq 1 \ \wedge\ 0 \leq y \leq 1 \ \wedge\ x \approx y \ \rightarrow\ x^2 \approx y^2$$

We rewrite this conjecture by reformulating y as $x+\varepsilon$ for some infinitesimal ε

$$\forall x, \varepsilon \in^* \mathbb{R}.\ 0 \leq x \leq 1 \ \wedge\ \varepsilon \approx 0 \ \rightarrow\ x^2 \approx (x+\varepsilon)^2$$

and then rewrite the conclusion to

$$\begin{aligned} x^2 &\approx (x+\varepsilon)^2 \\ &\approx x^2 + 2 \times \varepsilon \times x + \varepsilon^2 \\ &\approx x^2, \end{aligned}$$

since ε is infinitesimal and x is finite as it is in the range ${}^*[0,1]$ [2].

The techniques involved in this proof are of the sort employed by students when first confronted with calculus. Until non-standard analysis was formalised, these techniques were not sound and could lead to problems. Now it is possible formally to say that two numbers are very close together, and it is also possible to properly define what adding an infinitesimal amount actually means. For a good introductory article outlining all of the important issues in non-standard analysis see [Simpson, 1990]. Other texts, such as [Goldblatt, 1991, Robinson, 1966, Hurd and Loeb, 1985, Hoskins, 1990], providing a more complete coverage of the subject, are also of interest.

2.5 Summary

We have presented various important mathematical concepts which we use in our research. We have shown different approaches to formalising a theory of non-standard analysis, and given

[2] x must be finite in order to guarantee that $\varepsilon \times x \approx 0$

brief descriptions of formal constructions of both the reals and the hyperreals using axioms from a set theory. The choice of whether to introduce the field axioms, or to derive them from a more fundamental logic, will become important in the next chapter, when we present mechanised versions of both standard and non-standard analysis.

In sections 2.4.2 and 2.4.3 we gave simplified non-standard definitions for limit and continuity, which will be important later in the thesis when we study proofs from real analysis using such simplified definitions.

Chapter 3

Automated Theorem Proving

In this chapter we describe the work that exists in theorem proving in nalysis. The initial research in this area done by [Bledsoe et al., 1972], focussed on reasoning techniques for standard analysis, which were later applied to proofs in non-standard analysis. We review some of the significant work that has taken place since then on theorem proving analysis and non-standard analysis in both interactive and automatic settings. In each case, we state whether the work done provides an axiomatisation as its basis, or whether it develops a theory of analysis from a more fundamental logical perspective such as the approach of [Harrison, 1998] and [Fleuriot, 2001a].

In the final part of the chapter we give an overview of the central ideas of proof-planning, and briefly describe some of the implementational additions we have made to the $\lambda Clam$ system, which is the vehicle for analysing the structure of proof in non-standard analysis. We also show diagramatically how the architecture of $\lambda Clam$ is organised to allow us to automatically construct plans of proofs in which we are interested.

Throughout this section we use the symbol $\Rightarrow$ to denote rewriting, and $\rightarrow$ to denote implication. Conditional rewrites rules are written as $A \rightarrow \ B \Rightarrow C$, where A is the condition under which B rewrites to C. It is important the rewriting $(\Rightarrow)$ is not read as implication $(\rightarrow)$. See section 4.2.3 for an explanation of this. We use the turnstile symbol, $\vdash$, to denote derivability.

3.1 Theorem proving in analysis

In 1972, Bledsoe set a series of challenge problems to the theorem proving community, and together with Boyer, went about trying to prove them [Bledsoe et al., 1972]. The challenges he

set now form a corpus of examples [Bledsoe, 1990], and his first ideas are still being used by the latest theorem provers [Benzmüller et al., 1997].

3.1.1 Mechanised standard analysis

In this section we review some of the important work that has been done in mechanising proof in standard analysis.

Bledsoe and Boyer

The success of the impressive work done by Bledsoe and Boyer relies mainly on what they called their "Limit Heuristic". Although the prover was comprised of many methods, this was the part which was designed more specifically for limit theorems. Given a goal of a general form $|B| < E$, with a set of hypotheses containing one of the form $|A| < E'$, one tries to express B in the form $KA+L$. One then proves the following three subgoals for some M:

$$|K| < M$$
$$|A| < \tfrac{E}{2M}$$
$$|L| < \tfrac{E}{2}.$$

Two common theorems that we discuss further in chapter 5 are LIM+ and LIM$\times$, which pertain to real-valued functions. LIM+ states the sum of the limits equals the limit of the sums, and LIM$\times$ states that the product of the limits equals the limit of the products. For example in proving LIM$\times$, the conclusion can be expressed as $|f(x)g(x) - L_1L_2| < E$, and one hypothesis is $|f(x) - L_1| < E'$. The conclusion can be rewritten as

$$|g(x)(f(x) - L_1) + L_1(g(x) - L_2)| < E.$$

Now the original B has been expressed in the form $KA+L$. Now by the limit heuristic there are the following three subgoals remaining (for some M):

$$|g(x)| < M$$
$$|f(x) - L_1| < \tfrac{E}{2M}$$
$$|L_1(g(x) - L_2)| < \tfrac{E}{2}.$$

The proof then follows from the hypotheses in a much simpler way.

ΩMEGA

More recently there have been several other attempts to automatically prove many of these limit theorems, most notably by [Beeson, 1998] and [Melis, 1998, Melis, 1996]. In the ΩMEGA proof-planner and mathematical assistant, there has been much work done in developing proof-plans for limit theorems using what is referred to in [Benzmüller et al., 1997] as "multiple strategies". The ΩMEGA proof-planner builds on Bledsoe's work by introducing a "complex estimate", and a constraint solver in order to calculate the instantiations of the variables. A good example to illustrate this type of reasoning is the theorem $LIM+$. The difficult part of this proof in a standard setting is trying to find the appropriate instantiations for the δ term in the conclusion, and the ε terms in the hypotheses. In a rewriting setting, this means knowing the rule [1]

$$X+Y<E \;\Rightarrow\; X<\tfrac{E}{2} \wedge Y<\tfrac{E}{2}.$$

Melis argues that in most mathematical texts, the fact that the ε terms in the hypotheses are both instantiated to half the ε term in the conclusion, and that the δ term in the conclusion is chosen to be the minimum of the δ terms in the hypotheses, is plucking an answer out of nowhere [Melis, 1996]. This makes the proof very difficult to automate. The solution is to make all of these problematic variables meta-variables, and to create a partial proof-plan, which is completed by finally instantiating these variables using some form of constraint satisfaction.

This technique creates an abstraction of the actual proof, hence creating a partial plan π which is a tuple $(T,\prec,B,C)$, where T is a set of methods, $\prec$ is a partial order on T, B is a set of binding constraints on variables, and finally C is a set of inequalities. The set C of inequalities are passed to the constraint solver in order to instantiate the variables listed in B. The plan is set up with $T=t_0,t_\infty$, where t_0 are the hypotheses, and t_∞ is the final goal and $t_0 \prec t_\infty$. The planner refines the partial plan by introducing steps and constraints into the partial plan. The plan is complete when the initial state is transformed into a state where the goal holds and the constraints are satisfied. The general techniques used in solving standard $\varepsilon-\delta$ proofs are encapsulated in the two methods which are used specifically for solving inequalities. One uses such ideas as the triangle inequality, and the other instantiates variables in inequalities by introducing unifying substitutions. The work of the ΩMEGA group with regard to proof-planning is thoroughly covered in [Benzmüller et al., 1997, Melis, 1998, Melis, 1996]. This work is very successful in that it is able to prove many complicated limit theorems, but the

[1] Logical implication is in the opposite direction to rewriting

system described here has been used exclusively for limit theorems.

Mathpert

The Mathpert system is also capable of performing some automatic $\varepsilon - \delta$ proofs. [Beeson, 1998] presents how the *Weierstraß* component of Mathpert is capable of this sort of reasoning. Mathpert is a combination of both a formal system, and a system that can perform mathematical computations. He explains how some of the algorithms used cannot ensure soundness as can be done in a formal logical system. He presents a technique for solving $\varepsilon - \delta$ proofs which as in Melis' work involves introducing meta-variables and delaying their instantiation. The system combines computation with first order logic, by introducing certain additional features. These include a component for finding upper and lower bounds, which Beeson claims is as good as a very good calculus student at finding such bounds; a component for factor bounding, which proves that products of variables are small, if one is small and one has a bound; a component for exploiting the transitivity of like inequalities; and finally he allows the use of the mean value theorem, which splits certain goals into logically simpler subgoals. Beeson presents various proofs of continuity of specific functions in Mathpert using these techniques and a factoring algorithm common to the whole Mathpert system.

This work is intended to serve as a mathematical assistant for those working with $\varepsilon - \delta$ proofs. The system does not serve as a theorem prover, as some of the algorithms are not guaranteed to be sound. Its behaviour is thus more akin to that of a computer algebra system rather than a theorem prover. Moreover, many of the techniques that are available to the system during these proofs are quite advanced in themselves. Beeson himself says that when proving the continuity of $\sqrt{x}$, Mathpert uses the mean value theorem, which is a more complicated theorem than the continuity of $\sqrt{x}$. His reason for using the mean value theorem is that it is a less ad hoc rule than the factoring rule needed to otherwise complete the proof. This demonstrates the fact that the *Weierstraß* system is predominantly a mathematical assistant as opposed to a theorem prover.

PVS

The PVS system [Owre et al., 1992] includes an axiomatisation of the reals [Dutertre, 1996]. PVS is a specification and verification system designed to make formal proofs practical and applicable to real world problems, in particular software engineering. The axiomatisation of the real numbers has been included to verify properties of *hybrid systems* [Henzinger et al., 1997,

Henzinger et al., 1992]. In such systems physical constraints involve continuous functions of time. Given common results from analysis about continuous functions, reasoning about these functions of time becomes much simpler. The PVS interactive proof checker uses a classical higher order logic, with a rich type system that supports subtyping and dependent types. The libraries provided by Dutertre consist of analysis with *rational* functions, which is to say that they are comprised of identity functions, constants and the symbols $+, \times, -$ and $\backslash$. Also included are many theorems from the theory of real analysis, including most significantly results about the composition of continuous functions. As an example [Dutertre, 1996] shows a proof of the mean value theorem, given Rolle's theorem.

This implementation in PVS assumes many results from analysis, so that quick progress can be made on verification problems from hybrid systems. In [Gottliebsen, 2000], the axiomatisation is further used to construct proofs about transcendental functions. First a definition of partial sums is constructed, and then using the convergence theorems from the library provided by Dutertre, some trigonometric identities are proved. Also a continuity checker is implemented which uses some of the continuity lemmas from the libraries to determine whether more complicated compositions of functions are continuous. In later work, a study of definite integrals is performed in PVS using VSDITLU, a verifiable symbolic definite integral table look-up [Adams et al., 1999].

Coq

Another axiomatisation of real analysis has been done in the Coq system by [Cruz-Filipe, 2002] and [Mayero, 2001]. The former provides a constructive basis for many proofs, including Rolle's theorem, given results from power series; the latter is a generous axiomatisation which is used to give a classical proof of the three gap theorem which is a challenging problem for proof assistance systems.

HOL and Isabelle/HOL

The systems described above all use various axiomatisations in order to reason about analysis proofs. However, in both the HOL and Isabelle/HOL theorem provers, developed at the University of Cambridge, the methodology is one of definition rather than postulation. Both systems provide solid logical constructions of the real numbers. Jutting's translation of Landau's "Grundlagen der Analysis" in Automath is the first example of a rigorous mechanised definition of the real numbers [van Benthem Jutting, 1977]. This has been done by [Harrison, 1998]

in HOL using Cantor's method, and by both [Harrison, 1992] in HOL and [Fleuriot, 2001a] in Isabelle/HOL using Dedekind cuts. Standard analysis proofs have been performed in these interactive settings, and Harrison explains succinctly why this type of proof is difficult to automate [Harrison, 1998]:

> Very often one sets out to establish some overall bound on ε, say, and to get this one instantiates other ε-δ properties and uses the triangle law to get the result. The required instantiations generally follow not just from the fact to be proved, but from the structure of the intended proof. Taking the finished proof for granted, the reasoning is not deep, but it's often difficult to guess the right instantiations until the proof structure has been developed.

3.1.2 Mechanised non-standard analysis

Interestingly a significant proportion of those people who have worked on theorem proving in standard analysis have subsequently tried their hand at non-standard analysis. A brief overview of some of the work done in this area is given below. The majority of the work uses what is referred to here as an axiomatisation. This means that the object logic of the theorem prover has been augmented by the addition of various rules which have not been defined in terms of the axioms of the logic. [Fleuriot, 2001a] uses the HOL methodology of definition rather than postulation, and defines everything with respect to the axioms of higher order logic. His work does not therefore add any axioms to the axioms of the underlying logic, and hence does not use what is referred to here as an axiomatisation, apart from that of the higher order logic itself and some simple definitions for the construction of sets [Gordon and Melham, 1993].

Bledsoe and Ballantyne

[Bledsoe and Ballantyne, 1977] presents a resolution theorem prover whose proof engine is modified from an existing prover. The goal of this work is to automatically prove a substantial amount of a significant mathematical theory, and also to demonstrate how non-standard analysis allows mathematicians to prove theorems in analysis more easily than by using the $\varepsilon - \delta$ formulation. The paper presents the methods employed in the prover, and explains how it handles the various types that occur in non-standard analysis. Their work uses a generous axiomatisation and the prover is able to prove simple results about standard parts of numbers, as well as more significant results such as $LIM+$, and even the Bolzano-Weierstrass theorem that any bounded sequence has a convergent subsequence. By including non-standard definitions of *compactness*, *sequence convergence* and *continuity*, the domain of theorems proved is very

impressive.

Mathpert

Beeson has used non-standard analysis in the *Mathpert* system to great success. His work uses non-standard analysis to ensure the correctness of calculations, specifically in evaluating certain expressions which involve limits [Beeson, 1995]. His reason for appealing to non-standard analysis in this way is that by introducing infinitesimals one removes many of the problems in trying to prove tricky side conditions for limits. The example he gives is that in trying to calculate the derivative of the function $f(x) = \sqrt{x}$, where $f : \mathbb{R} \to \mathbb{R}$, it is necessary to add certain assumptions which involve variables which are bound by the limit term. The limit expression in question is given by the equation

$$\frac{d}{dx}(\sqrt{x}) \equiv \lim_{h \to 0} \frac{\sqrt{x+h} - \sqrt{x}}{h}.$$

Beeson explains how in the calculation of this limit term, one has to use the rule

$$y \geq 0 \to (\sqrt{y})^2 \Rightarrow y$$

but this causes problems because in the calculation of the above limit term one has to say that $(\sqrt{x+h})^2 = x+h$. In order to do this, the assumption that $x+h \geq 0$ has to be added to the system. This is not possible as h is bound within the limit term, and hence referring to it outside the term is futile. The solution is to simply add $x \geq 0$ as an assumption, but Beeson claims that doing this can only be done in a very ad hoc way. His solution is to axiomatise non-standard analysis, require h to be infinitesimal, and to write a specialised procedure for the elimination of infinitesimals from this type of expression. In other words he is trying to automate the kind of reasoning about limits that is performed in schools when differentiation is first introduced. He introduces the axiomatisation used, and explains in detail the algorithm developed for eliminating infinitesimals. He also demonstrates the sort of proofs that his system is capable of doing, and claims that Mathpert is able to solve any example of a limit problem found in calculus text books.

ACL2

ACL2 [Kaufmann and Moore, 1997] is the successor to the *NQTHM* (or Boyer-Moore) theorem provers [Boyer and Moore, 1990], and is capable of automatically proving a wide range

of theorems. Originally the system could not reason about irrational numbers, as it permitted a proof that the square root of two did not exist. [Gamboa, 1999] has used non-standard analysis to approximate results over the real numbers. He uses Nelson's internal set theory [Nelson, 1977] to axiomatise the non-standard numbers. He explains how irrational numbers have been left out of ACL2 in order to try and keep the prover as close to common lisp as possible. In ACL2 it is possible to prove that $\sqrt{2}$ does not exist, in the same way that $\sqrt{2}$ can be proved to be irrational. Hence adding irrational numbers to ACL2 by the addition of axioms would produce an inconsistent theory. Gamboa presents an iterative method for approximating the value of a square root, and then uses his axiomatisation in non-standard analysis to define a square root function and prove theorems involving it. Many of the lemmas needed to automate these proofs are given by the user. He also uses non-standard analysis to prove interesting versions of classical theorems of analysis such as the intermediate value theorem, and presents some definitions of transcendental functions such as *sin* and *cos*, and proves Euler's theorem: $e^{i\pi} + 1 = 0$. It is unclear exactly what degree of automation is involved in proving these theorems.

Bedrax

Bedrax has also written a theorem prover using non-standard analysis, which serves mainly as a mathematical assistant [Bedrax, 1993]. She uses a very generous axiomatisation, which is given as a set of inference rules which are not formulated within the underlying formal logic of the system. She is thus able to prove such theorems as the intermediate value theorem. This work is not as significant as that of Beeson, as the prover is both axiomatised and interactive, thus removing two of the most challenging aspects of theorem proving in the non-standard domain.

Isabelle/HOL

The most significant work done in the area of non-standard analysis and theorem proving has been performed in Isabelle/HOL by [Fleuriot, 2001a]. In this work, the hyperreals are constructed from just the axioms of the underlying logic of the theorem prover, and the construction used to prove substantial portions of real analysis in the hyperreals.

As in the case of Harrison's work [Harrison, 1998], the HOL methodology requires derivation of mathematical notions rather than postulating them, and so anything proved in this setting is guaranteed to be correct. Since everything is formulated within higher order logic the need

for axiomatisation is obviated. Axiomatisations can contain inconsistencies, as has been the case in the past for non-standard analysis. In order to construct the hyperreals in Isabelle, the reals must first be constructed. This was done by following the path $\mathbb{Z}^+ \longrightarrow \mathbb{Q}^+ \longrightarrow \mathbb{R}^+ \longrightarrow \mathbb{R}$. From the real numbers, the hyperreals are constructed by first defining the notion of an ultrafilter.

Fleuriot shows how he proves a version of the axiom of choice – Zorn's Lemma – to to be able to prove the existence of a free ultrafilter, which is central to the ultrapower construction of the hyperreals. This is possible in Isabelle/HOL because the Hilbert ε operator is used as a primitive in the axiomatisation of Higher Order Logic. He demonstrates his subsequent construction of the hyperreals, and the correctness of the construction with the respect to embedding the real numbers into the hyperreals.

The work then goes on to prove many important results, which are required when formulating a theory of calculus in non-standard analysis e.g. the standard part theorem, which states that all finite hyperreal numbers are infinitely close to a unique real number. Notions of limit and continuity are explained in the non-standard setting, and proofs of the equivalence of the standard and non-standard definitions are given. The work also shows that the non-standard definition of limit can be interpreted in a useful way when the limit is at infinity or at zero. Important calculus theorems are proved such as the chain rule and Rolle's Theorem. These proofs are just an indication of the sort of proof that Isabelle is capable of, using Fleuriot's construction of the hyperreals, and subsequent formalisation of a mechanised infinitesimal calculus.

More recently this work has been extended by using the existing framework to introduce transcendental functions such as *sin* and *cos*, and formalise the notion of power series in general [Fleuriot, 2000].

3.2 Proof-planning

Proof-planning [Bundy, 1988] is a technique for devising an overall plan for a proof, which can then be used to guide the proof search itself. A proof-plan consists of methods and critics. The methods embody common patterns within proofs, such as the use of induction, which correspond to tactics at the object-level, which carry out these methods explicitly. For each conjecture a precise proof-plan is built, but a general form of a proof-plan can be developed for certain types of conjecture. The mechanism for proof *critics* with respect to proof methods is described when Ireland's work on proof planning and the productive use of failure is presented in section 3.2.4.

The main advantage of proof planning is that it reduces the search space by reasoning using methods, which are a specification of when a tactic applies, and what its effects are. The λ*Clam* theorem prover [Richardson et al., 1998] developed at the University of Edinburgh, and the ΩMEGA system [Benzmüller et al., 1997], developed at the University of Saarbrücken, are examples of proof-planners which embody many of the notions developed in the field. In what follows, a brief overview of the main concepts needed for this work is provided.

3.2.1 Proof plans

Induction is a perfect example for demonstrating the use of proof-planning. An inductive proof always takes on a certain form, and by imposing a plan on the search, the choices to be made are considerably reduced. In general an inductive proof always contains a base case, which may itself involve subsequent inductions, and a step case, which introduces extra term structure to the conjecture. A typical proof-plan for induction would look like figure 3.1. This high level plan is intended as an abstraction of any inductive proof, and it has been very successful at proving several theorems about natural numbers and lists automatically. It may not at first be immediately obvious where the difference lies in this kind of plan, and a waterfall system. A waterfall of methods is an ordered set of methods which are tried cyclically until the goal is proved or no method applies. In λ*Clam*, however, the use of methodicals— constructs which comprise of many applications of other methods— allows the possible form of a plan to be more adaptable than a simple waterfall system such as the one in *NQTHM*. Using this system, it is easier to develop a generic abstraction of certain types of proof.

As described in [Ireland, 1992, Ireland and Bundy, 1996], a method in proof planning has a number of "slots" assigned to it. It has a name by which it is recognised, an input which is matched against the current goal, and a set of preconditions which determine whether the method is applicable to the current goal. It also has a set of effects which define what must be satisfied after the method has been applied, and a tactic which controls the object level prover. Finally it has an output which is the result of applying the effects to the current goal. When the object level prover is referred to, it means that an automated theorem prover that could carry out the tactic specified by the proof plan developed in λ*Clam*. An example of the application of critics demonstrating the nature of slots is given in section 3.2.4.

In the version of λ*Clam* that we use, there is a distinction made between *atomic* methods and *compound* methods. Those which have a form similar to that described above, where slots are used, are called *atomic* methods. and compound methods comprise of a sequence of

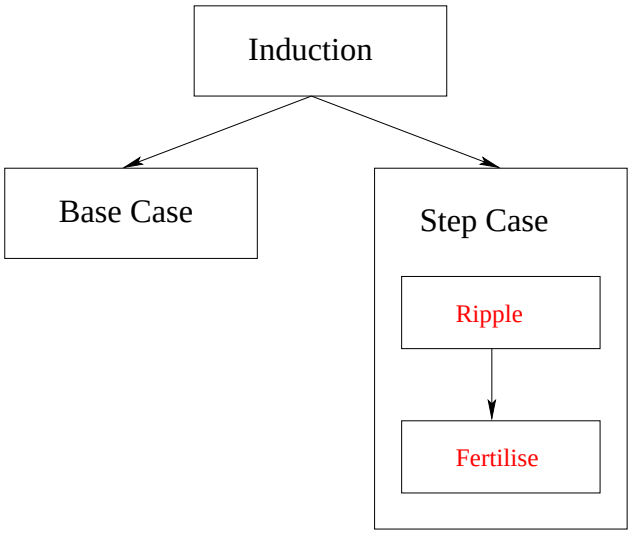

Figure 3.1: General purpose induction proof-plan

applications of atomic methods. For a description of how compound methods are composed see section 3.2.2.

3.2.2 Methodicals

Methodicals are a way of combining methods. Compound methods are a set of atomic methods combined using methodicals. A simple example of a methodical is `then_meth`, which applies one method, and then applies another to the resulting goals. The λ*Clam* methodicals which are used in the work presented in this thesis are the following.

id_meth

This automatically succeeds and is used as a way of passing goals back to higher level compound methods.

triv_meth

This succeeds if the goal is trivially true.

orelse_meth

This attempts one method, and if it fails, attempts another.

cond_meth

This only applies a method if a condition is passed.

try_meth

This attempts a method but does not fail if the method fails.

repeat_meth

This repeatedly applies a given method until it cannot apply.

then_meth

This applies one method, and then another to all the resulting goals. If one method solves a goal, then the branch is closed and the next goal is tackled.

then_meths

This applies one method, and then gives the opportunity to apply different methods to different resulting goals.

pair_meth

This method is used in conjunction with `then_meths` and applies a pair of methods to a pair of goals.

patch_meth

This methodical tries to apply an atomic method and then uses a meta-interpreter to analyse which of the preconditions to the atomic method failed, using a critic strategy to suggest a patch.

3.2.3 Backtracking

λ*Clam* is written in λprolog [Miller and Nadathur, 1988], which is a higher order declarative implementation of prolog. When a call to a λprolog predicate fails, backtracking occurs to the nearest choice point in a depth-first fashion. In λ*Clam*, backtracking can occur during the applications of methods by the **orelse_meth** methodical, and also in the preconditions to atomic methods. In the case of the induction proof-plan shown in figure 3.1 for example, backtracking can occur in the choice of rule to apply in symbolic evaluation and rippling.

As an example of backtracking consider the next situation, where we assume. Here we are assuming a depth-first planning strategy, which has available to it the following rewrite rules:

$$s(X)+Y \Rightarrow X+s(Y) \tag{3.1}$$

$$s(X)+Y \Rightarrow s(X+Y). \tag{3.2}$$

We have to establish a proof for the conjecture

$$x : \mathbb{N},\ y : \mathbb{N} \vdash s(x) + y = s(x + y).$$

λ*Clam* tackles this with the following compound method:

```
(then_meth
      (repeat_meth rewriting)
       reflexivity)
```

where `rewriting` is a general rewriting method, and `reflexivity` is a method which looks for goals of the form $X = X$. When it applies the compound method to the goal, λ*Clam* repeatedly applies rewrite rules until no further rewriting is possible. This yields the goal

$$x : \mathbb{N},\ y : \mathbb{N} \vdash x + s(y) = s(x + y)$$

which is not an identity. Now the planner backtracks to the last choice point, where it applied rule (3.1). The rewrite rule (3.2) then applies, and the resulting goal is an identity and so the proof-plan succeeds.

3.2.4 The productive use of failure in proof-planning

The use of inductive theorem provers such as *NQTHM* shows that failed proofs can provide useful information about failure, and hence can help to suggest ways to render proofs successful. One piece of useful information that could come out of a failed proof attempt is a suggestion of how to pre-process the conjecture in some way to help the proof succeed. In proof-planning this information is captured by *critics*. This is of particular importance to the work presented in this thesis, as we make use of critics throughout.

The specification of a critic is comprised of a number of slots in the same way as that of a method. The definition given by [Ireland, 1992] is:

```
critic(Name(Arguments),
       Input,
       Preconditions,
       Effects,
       Output,
       Tactic).
```

Here the name slot is the name of the critic, which corresponds to the method name to whose failure it is reacting. The input slot holds the current goal sequent. The preconditions slot contains the preconditions as to whether the critic should fire. The effects slot is used to evaluate the new subgoal sequents. The output slot posts the new subgoals, and the tactic slot specifies the object-level tactic.

Consider the following attempted proof of a simple equational conjecture in which a critic is able to react to a common failure pattern. The conjecture to be proved is

$$(x+y)^2 = x^2 + (x \times y) + (y \times x) + y^2$$

where we have the following rewrite rules available to us:

$$\begin{aligned} X^2 &\Rightarrow X \times X \\ (X+Y) \times Z &\Rightarrow (X \times Z) + (Y \times Z) \end{aligned}$$

The only methods that are available to the planner are:

`rewrite_with(Rule)` This method rewrites the current goal with the rewrite rule specified by `Rule`;

`identity` This method checks to see whether the current goal is an identity, i.e. something of the form $X = X$ and completes the proof.

The critic is attached to the `rewrite_with` method, and so reacts to its failure, and suggests a new rule according to the difference between each side of the equality. The preconditions of this critic are simply that no rewrite rule can apply, and the current goal is not an identity. The patch slot of the critic must then prove two subgoals. The original conjecture must still be proved, and the rule suggested must also be proved to be sound.

The proof proceeds until the point

$$(x \times (x+y)) + (y \times (x+y)) = (x \times x) + (x \times y) + (y \times x) + (y \times y)$$

when no more rewrite rules apply. The critic then fires, as the rewriting method has failed. It analyses the failure of the rewriting method, and tries to suggest a rule that will reduce the difference between the two sides of the equality. The sensible rule to suggest in this situation is

$$(X \times (Y+X)) \Rightarrow (X \times Y) + (X \times X)$$

which completes one branch of the proof. The proof that the above rewrite rule is sound is omitted here. Critics can be used in many situations to complete otherwise stuck proofs. Their attraction is that they act in a way that humans do when they perform mathematical proof. When a proof is carried out, it is not a sensible strategy simply to give up once a certain path is blocked; the reason for the blockage should first be analysed. In this very simple case, the reason is not very complicated, but the critic is able to identify which term is not being rewritten, and suggest a sensible rewrite rule which will complete the proof.

3.2.5 Rippling

Rippling is a heuristic used in proof-planning for guiding the proof search. It was initially motivated by Aubin's observation on how terms introduced by induction are affected by rewriting [Aubin, 1976, Aubin, 1979]. Bundy formalised this idea into a theory of annotated rewriting [Bundy et al., 1993], and a formal calculus has been developed from which one can prove termination [Basin and Walsh, 1996]. The idea of rippling has been extended to non-inductive settings in [Yoshida, 1993, Yoshida et al., 1994, Walsh et al., 1992, Hutter, 1997]. A full worked example using rippling can be seen in section 3.2.7.

Rippling provides a formal way of annotating conclusions of a conjecture in such a way that the difference between the conclusion and the hypotheses are represented by what are known as "wave fronts" and "wave holes". For example the following conjecture has been annotated according to the rules of rippling; the hypothesis is contained within the conjecture. Those terms in the conclusion which do not appear in the hypothesis are contained in the shaded parts of the wave fronts; the wave holes are the non-shaded parts enclosed within the wave fronts:

$$x = y \vdash \boxed{s(\boxed{x})}^{\uparrow} = \boxed{s(\boxed{y})}^{\uparrow}.$$

Rewrite rules are also annotated according to some preservation and measure reducing rules. When annotated the rewrite rules are referred to as "wave-rules". As rewriting takes place, this annotation changes according to the annotation on the wave rules. The advantage of using rippling is that termination is guaranteed, and one can either determine how close a conjecture is to being proved, or the reason for its failure by analysing the annotation. If the proof is successful, then once no more rewriting applies, the terms that are contained in the wave-holes should be instances of the hypotheses. Completing a proof by this sort of instantiation is an example of "fertilisation" [Bundy, 1988], which is described further in section 3.2.6.

As just mentioned, rippling can be shown to be terminating. This is done by imposing a measure on the annotated term and showing that it decreases. One can ripple *out*, *in*, or

sideways. In all of the examples we present in this thesis, we ripple out, which is signified by an up arrow on the wave front. Intuitively this means that as rippling proceeds, more term structure in the conclusion becomes encapsulated by the wave front. If this is not possible, then one can ripple sideways and then the arrow on the wave front points downwards, and we start to ripple in. Intuitively, this means that the annotation decreases until any extra term structure that exists in the conclusion in comparison with the hypotheses exists in the same position as a universally quantified variable in one of the hypotheses. A term in such a position is known as a *sink*, and rippling in cannot be successful unless a sink exists in the conclusion. Sinks are shown in this presentation of rippling by a light yellow background surrounding the sink itself. A typical example of such a rippling process is when proving theorems about tail-recursive functions, as described in [Ireland and Bundy, 1996]. Rippling sideways can only change the direction from out to in, ensuring termination.

An extension of rippling is *coloured rippling* [Yoshida, 1993] which extends the notion of rippling to cases where more than one hypothesis can represent wave holes in the conclusion. This extends to non-inductive theories where multiple hypotheses exist. The reason why it is called *coloured* rippling is that a colour is attributed to each hypothesis, and annotated as such in the wave holes of the conclusion. For example we could write the conjecture:

$$a \approx b, c \approx d \vdash \boxed{a + \boxed{c}^{\uparrow}} \approx \boxed{b + \boxed{d}^{\uparrow}}^{\uparrow}$$

and then introduce an annotated rewrite rule:

$$\boxed{a + \boxed{c}^{\uparrow}} \approx \boxed{b + \boxed{d}^{\uparrow}}^{\uparrow} \Rightarrow \boxed{a \approx b + \boxed{c \approx d}^{\uparrow}}^{\uparrow}$$

which allows us to rewrite the conclusion to the point where the hypotheses apply. Introducing this sort of annotation allows us to keep track of the terms from each hypotheses in the conclusion during the rewriting process.

More generally, the notion of *embeddings* has been developed to account for rippling in a higher order setting. [Smaill and Green, 1996] describes how the λ*Clam* proof-planner uses embeddings to enable hypotheses to embed in conclusions in the presence of lambda terms. Importantly a measure has been devised for embeddings which preserves the termination properties of rippling. Coloured rippling can be described in this context by attributing an embedding to each hypothesis. In λ*Clam*, the method which incorporates the ideas of rippling with embeddings is called the `wave` method.

Work by [Hutter and Kohlhase, 1997] extends the idea of annotation by working on higher order terms, and by annotating equational conjectures according to the difference between

the expressions on each side of the equality [Hutter, 1997]. He refers to this annotation as "colouring terms" although the general idea behind the annotation remains the same.

The full calculus of rippling is too complicated to explain fully in this section. For a full description of such a calculus see [Basin and Walsh, 1996], and for a detailed description of Hutter's work on colouring terms for equational rewriting see [Hutter, 1997]. As an example of a typical proof using annotated rewriting see the example proof of a non-standard conjecture using proof-planning and annotation in the next section.

3.2.6 Fertilisation

Fertilisation is a proof-planning technique for completing a proof-plan. There are two main forms of fertilisation which can take place: *strong* fertilisation and *weak* fertilisation. An example of strong fertilisation can be seen in section 3.2.7.

Strong fertilisation corresponds to instantiating the hypotheses with the conclusion and hence completing the proof. For example, the goal

$$a \approx b, c \approx d \;\vdash\; \boxed{a \approx b \wedge c \approx d}^{\uparrow}$$

can be strongly fertilised since each of the conjuncts in the conclusion, which exist in wave holes, correspond to an instance of the hypotheses.

Weak fertilisation takes place when strong fertilisation cannot be performed, but there is a way of rewriting the conclusion with the hypotheses. This happens if there is a hypothesis which involves an equality, an equivalence or an implication. In the case of implication, the issue of polarity must be taken into account. This means that the corresponding rewrite rule rewrites in the opposite direction to the implication.

In some cases we need to use *piecewise* fertilisation, which analyses the failure of any sinks to match. For example consider the goal

$$R(t), \forall x \in \tau.\, P(x) \rightarrow Q(x), P(y) \vdash \boxed{P(z) \rightarrow Q(y) \wedge R(t)}^{\uparrow}$$

to which strong fertilisation does not immediately apply. In order to determine that this goal is provable, piecewise fertilisation first notes that the term in the red wave hole can immediately be discharged as it corresponds to a hypothesis. The blue wave hole contains mismatching sinks, and so does not correspond to an instantiation of a hypothesis. We note that the term $Q(y)$ matches with the conclusion of the implication of the universally quantified hypothesis, and so if $P(y)$ can be established then the whole goal is provable.

3.2.7 Worked example

As an example of rippling and fertilisation in action, we show a proof of the associativity of $+$ over natural numbers. We state the theorem as

$$\vdash \forall x,y,z:\mathbb{N}.\ \ (x+y)+z = x+(y+z).$$

Induction on x is chosen setting up the induction hypothesis

$$x:\mathbb{N},\ \ \forall y,z:\mathbb{N}.\ \ (x+y)+z = x+(y+z)$$

and the step case conclusion

$$\vdash\ \ (\,s(x)^{\uparrow}+y)+z = s(x)^{\uparrow}+(y+z).$$

Notice here the existence of sinks corresponding to the position of universally quantified variables in the hypotheses. The base case becomes

$$\vdash \forall y,z:\mathbb{N}.\ \ (0+y)+z = 0+(y+z).$$

We solve the base case using the rewrite-rule

$$0+X\ \Rightarrow\ X.$$

In order to solve the step case, we use wave-rules

$$s(X)^{\uparrow}+X\ \ \Rightarrow\ \ s(X+Y)^{\uparrow}$$
$$s(X)^{\uparrow} = s(Y)^{\uparrow}\ \ \Rightarrow\ \ X = y$$

Apply the first of these repeatedly reduces the annotated goal to

$$s(\,(x+y)+z\,)^{\uparrow} = s(\,x+(y+z)\,)^{\uparrow}.$$

Then applying the second yields

$$(x+y)+z\ \ =\ \ x+(y+z)$$

which is an instantiation of the induction hypothesis. At this point strong fertilisation applies and the proof-plan is complete.

3.2.8 Proof Architecture

$\lambda Clam$ uses all of the above techniques to construct proof-plans. We use a depth first planner for our work, and so it is important to note where and how backtracking can occur. We note the important places where choices can be made in constructing a proof-plan.

Method choices

When a method with a certain name is invoked by a proof-plan, there may be more than one clause for that method.

Critic choices

When a critic fires, the pattern of failure may be recognised by more than one critic, in which case more than one critic is attempted.

Choices within atomic methods and critics

The most common form of backtracking occurs within the meta-language of the atomic method and critic definitions. These choice points do not show up in the proof-plan itself, as the proof-plan is composed of atomic method applications, but significantly affect the performance of the system.

Methodical choices

When an `orelse_meth` is employed in a compound method, backtracking can occur at this point.

The way in which backtracking occurs in the presence of critics is vital. If a critic is attached to a method, then it will fire as soon as the method fires, not allowing the method to backtrack to other possible method applications. The methodical language allows us to construct a solution to this problem by using the `try_meth` methodical. For example, if we want to be able to attach a critic to the `wave` method, then we would write

```
(patch_meth wave wave_critic_strat)
```

but this reacts to the first failure of the wave method. If this is not the desired behaviour, then we can write

```
(orelse_meth
  (try_meth wave)
  (patch_meth wave wave_critic_strat))
```

which will first try all of the occurrences of the wave method, and if this fails then try the patch to the failure.

3.3 Summary

In this chapter we have presented the important work done in mechanising proof in analysis, and outlined the main techniques used in proof-planning. Throughout the thesis we will refer to the proof-planning terminology such as methods, critics and methodicals whenever appropriate.

[Bledsoe and Ballantyne, 1977] presents work on proving real analysis theorems using an axiomatisation for non-standard analysis and a resolution style theorem prover. This work presents fully automated proofs of some of the same conjectures we study later in chapter 5. [Gamboa, 1999] presents a proof of the Intermediate Value Theorem, which is pertinent to our work, which we present in chapter 6. [Fleuriot, 2001a] provides a mechanised construction of non-standard analysis, and proves many of the theorems we study in our work. This is important as it form the formal basis for the axiomatisation we present in the next chapter.

Chapter 4

Conceptual Framework

In this chapter we discuss the methodology behind the research. We show our approach, explicitly stating how we arrive at planning proofs of theorems automatically. We first describe in detail the goals of the work, and the techniques we employ to achieve these goals. We go on to describe our logic and axiomatisation, citing explicitly each axiom. We describe some theoretical issues regarding proof in the non-standard domain, and finally we give a set of evaluation criteria by which the success of the work should be judged, and discuss some of the issues regarding evaluation.

4.1 The methodology

We intend to show that proof-planning can encapsulate the common pattern of reasoning in non-standard analysis proofs. We show that there is a common structure to reasoning in non-standard analysis, and also that proof-planning is a useful tool for discovering such structure in proof.

4.1.1 The research hypothesis

We state our research hypothesis as:

> Through proof-planning we arrive at intuitive and successful representations of the structure of proof in non-standard analysis.

By successful here we mean that we are able to yield proof-plans for the theorems we study, and by intuitive we mean that the reasoning patterns involved follow what we would expect when

the proofs are performed by hand. We test this hypothesis, using the information gathered from the evaluation measurements outlined in 4.4.3.

4.1.2 Research goals

We construct a set of plan-specifications which will account for many theorems, stated in non-standard terms. We hope that these plans will lead us to an understanding of the structure of proof in non-standard analysis. We start from an axiomatisation of non-standard analysis whose axioms are in fact theorems of a more fundamental logic, namely the higher-order logic included in the Isabelle/HOL theorem prover [Paulson, 1989].

In some text books on non-standard analysis (for example [Keisler, 1977]), it is often claimed that the reasoning involved in non-standard analysis matches that of Newton and Leibniz, in their non-rigorous early calculus proofs. We show that the reasoning involved in the proofs we study follows similar patterns, given the axiomatisation we have in $\lambda Clam$, by evaluating our results according to the criteria set out in section 4.4.1. We further demonstrate, by using non-standard analysis as a case study, the suitability of proof-planning to automated theorem proving.

4.1.3 Proof-planning

We need to set out precisely what our framework is for this research. We first explain some nomenclature, and then give a precise explanation of our proof-planning structure.

Nomenclature for proof-plans

It is important to make a distinction between the type of proof-plan which is given to the planner, and the proof-plan output by the planner.

The proof-plan which is given to the planner, consists of applications of methods and critics, and hence is expressed in the meta-logic. It consists of general reasoning techniques, expressed as methodicals, which apply to methods. This sort of proof-plan, which exists at the meta-level, is of type `method`. This can be seen as a specification of a proof strategy for an object level prover. We will refer to this type of proof-plan as the *plan-specification*.

The proof-plan which results as output from the planner consists of a tree of tactic applications, as expressed by the *tactic* slot of the methods involved. In fact this proof-plan is of type `tactic`. This sort of proof-plan therefore exists at the object level. From now on, when we refer to *proof-plan*, we mean this sort of plan.

The proof-planning framework

In our research we are predominantly interested in the structure of the proof at the method level. We do not claim to execute the proof-plans in an object-level theorem prover, although this has been done for some combinations of proof-planning systems and object level provers such as for example [Boulton et al., 1998, Bundy et al., 1990] and [Castellini and Smaill, 2002]. As methods are user-defined, it is possible for a method to be unsound, if for example there is a mistake in its definition. It is also possible that a method may not correspond to a set of tactics at the object level. We strive for soundness, which depends upon the following issues:

1. the existing mechanisms for rewriting, and method application are sound in $\lambda Clam$;
2. the extra inference rules and axioms which comprise the theory we add to the system are sound;
3. the machine representation of these axioms and inference rules is correct.

By formally stating our framework in section 4.2, we show exactly the rules that can make up the effect of a method on a proof state. This means that we do not encounter problems with the second soundness issue.

4.1.4 Testing

Our testing methodology is to separate the set of theorems into a test set and a development set, and to judge both sets according to various evaluation criteria using quantitative experiments.

We create a set of plan-specifications by analysing the proofs of the theorems in the development set. These examples are intended to be indicative of the different types of theorems that one might encounter in non-standard analysis. Once an initial set of plan-specifications has been found we reduce their number by grouping them, and accounting for each group by a more general plan-specification. For example, consider the following two schematic plan-specifications:

```
1. (repeat_meth                  2. (then_meth taut
       (then_meth trivial              (then_meth (repeat_meth sym_eval)
       (then_meth taut sym_eval))          trivial)
```

Here `taut` is a tautology method which either discharges the goal or leaves it alone, and `sym_eval` represents symbolic evaluation. Plan-specification `2` is a more specific version of example `1`. Any resulting proof-plan produced by `2` could equally be achieved by `1`. In this case we only include `1` in the set of plan-specifications. See section 4.4.2 for a discussion of this procedure.

Once we have a reduced set of plan-specifications, we apply each one to each of our test examples, and evaluate their success according to the criteria set out in section 4.4.1.

In order to illustrate the development process we document a number of facts for each of the examples in the development set. Our approach follows partly the evaluation methodology outlined in [Cantu et al., 1996] and considers:

- The time taken for the planner to find a complete plan for the theorem;
- The time taken for the implementation of each of the methods and critics and ancillary code;
- The number of lemmas used in the plan.

These facts are documented because they demonstrate how successful the development process was, and how well the development set was chosen.

4.2 Formal Framework

One vital part of the framework in which the proof-plans are executed is the logical axiomatisation adopted. In this section we first review our logical framework, and then introduce all the types and subtypes used in $\lambda Clam$. We introduce our constant symbols and our axioms for both standard and non-standard analysis.

4.2.1 Logic

The basic logic used is higher-order and is represented by a simply typed sequent calculus with single conclusion. The sequent rules which the planner has at its disposal are given in table 4.1. Here, $\top$ denotes truth, and $\bot$ denotes falsehood. Importantly we introduce the type $\mathbb{B}$ to which $\top$ and $\bot$ belong, and the constant symbol $=$ with polymorphic type $X \times X \rightarrow \mathbb{B}$, which we use in the substitution rule *eq*. We assume the standard transformation rules for the lambda-calculus, and assume a type system similar to Church's formulation for the simple theory of types [Church, 1940] augmented with a product type. In the presentation of our

sequent calculus shown in table 4.1, the contexts in the sequents (Shown as Γ) are lists of typing judgements and propositional-typed formulae. The structural rules for adding formulae or typing judgements to the context are omitted here as list constructors are used internally in $\lambda Clam$. Any extra elements in the context other than Γ can be thought of as singleton lists, and the , which appears in the context as an append function.

$$\frac{}{\Gamma, A \vdash A}\ \text{axiom} \qquad \frac{}{\Gamma \vdash \top}\ \text{axiom}$$

$$\frac{}{\Gamma, \bot \vdash A}\ \text{f\_axiom} \qquad \frac{\Gamma, A \vdash D \quad \Gamma \vdash A}{\Gamma \vdash D}\ \text{cut}$$

$$\frac{\Gamma \vdash A \quad \Gamma, B \vdash D}{\Gamma, A \to B \vdash D}\ l{\to} \qquad \frac{\Gamma, A \vdash B}{\Gamma \vdash A \to B}\ r{\to}$$

$$\frac{\Gamma, A, B \vdash D}{\Gamma, A \wedge B \vdash D}\ l{\wedge} \qquad \frac{\Gamma \vdash A \quad \Gamma \vdash B}{\Gamma \vdash A \wedge B}\ r{\wedge}$$

$$\frac{\Gamma, A \vdash D \quad \Gamma, B \vdash D}{\Gamma, A \vee B \vdash D}\ l{\vee} \qquad \frac{\Gamma \vdash A}{\Gamma \vdash A \vee B}\ r{\vee} \qquad \frac{\Gamma \vdash B}{\Gamma \vdash A \vee B}\ r{\vee}$$

$$\frac{\Gamma \vdash A}{\Gamma, \neg A \vdash B}\ l{\neg} \qquad \frac{\Gamma, A \vdash \bot}{\Gamma \vdash \neg A}\ r{\neg}$$

$$\frac{\Gamma, A[c/x] \vdash D}{\Gamma, \forall x \in \tau.\ A \vdash D}\ l{\forall} \qquad \frac{a : \tau, \Gamma \vdash A[a/x]}{\Gamma \vdash \forall x \in \tau.\ A}\ r{\forall}$$

$$\frac{a : \tau, \Gamma, A[a/x] \vdash D}{\Gamma, \exists x \in \tau . A \vdash D}\ l{\exists} \qquad \frac{\Gamma \vdash A[c/x]}{\Gamma \vdash \exists x \in \tau . A}\ r{\exists}$$

$$\frac{B \vee \neg B, \Gamma \vdash A}{\Gamma \vdash A}\ em \qquad \frac{\Gamma \vdash P(x)}{x = y, \Gamma \vdash P(y)}\ eq$$

variable a cannot appear anywhere in the conclusion for $r\forall$ and $l\exists$

Table 4.1: The sequent calculus

4.2.2 Number types

The development of non-standard analysis as given in section 2.3.1 shows that one can construct non-standard extensions of all the basic number systems used in standard analysis. We concern ourselves here mainly with the natural numbers and the real numbers.

The naturals and hypernaturals

The naturals ($\mathbb{N}$) can be defined using the Peano axioms. We introduce a successor function, and a *zero* element, by which induction can be performed. We introduce a type `nat`, and a type `hypernat`, and relate them by introducing an operator `emb`, of type $\texttt{nat} \to \texttt{hypernat}$, which denotes the embedding of a natural number within the hypernaturals ${}^*\mathbb{N}$. As the set $\mathbb{N}$ within ${}^*\mathbb{N}$ is a copy of the natural numbers, it must preserve all of the properties of the natural numbers. `nat` and `hypernat` are the internal representations for the naturals and hypernaturals, but in this presentation we use $\mathbb{N}$ and ${}^*\mathbb{N}$.

The reals and hyperreals

In our work we introduce the real numbers and the hyperreal numbers as simple types `real` and `hyperreal`, and introduce the usual field axiomatisation. In addition to this we introduce some axioms which express the connection between these numbers system. As with the natural numbers we introduce the operators `emb` and `ext`, of type $\mathbb{R} \to {}^*\mathbb{R}$ and $(\mathbb{R} \to \mathbb{R}) \to ({}^*\mathbb{R} \to {}^*\mathbb{R})$ respectively. In the presentation of the axioms we write $\texttt{emb}(x)$ as $\widehat{x}$, and $\texttt{ext}(f)$ as ${}^*f$.

An important subtype of the hyperreals is the *finite* numbers. The real numbers are all finite, but there are also other finite hyperreal numbers, such as hyperreals which are not equal but infinitely close to real numbers. This subtype is important because multiplication is not uniformly continuous over the hyperreals. In other words, for any hyperreals x, y and z, such that $y \approx z$, it is not the case that $x \times y \approx x \times z$. This is however true if x is finite. Thus many rules involving multiplication only apply to finite hyperreals. The way this subtype is introduced is by predicate subtyping. We introduce a predicate $finite$ which takes a single hyperreal argument.

4.2.3 Rippling and rewriting

In $\lambda Clam$, axioms are expressed as rewrite rules. For example, the reflexivity axiom for equality is written:

$$\forall x : \tau . x = x$$

In $\lambda Clam$ this is expressed as a rewrite rule: $X = X \Rightarrow \top$.

The methods which use rewrite rules in $\lambda Clam$ are rippling and symbolic evaluation. Rippling and symbolic evaluation differ from a more standard notion of rewriting as expounded for example in [Baader and Nipkow, 1998], where rewriting refers to finding normal forms for terms. In $\lambda Clam$ we use a more general notion, where we do not only define rewrite rules from formulae whose outermost relation is equivalence or equality. We also define rewrite rules from formulae whose outermost relation is implication. The *polarity* of a term refers to which way a uni-directional rewrite rule applies when matched to the term. Stated simply, if all of the negation and implication symbols are removed using the sequent rules in section 4.2.1, then terms which appear on the left of the sequent have *negative polarity*, whereas terms which appear on the right have *positive polarity*. Terms can appear both on the right and the left of an implication, in which case each specific instance of the term has its own polarity.

When formulae with implications are used as rewrite rules, the direction in which they apply depends on the polarity of the term. We rewrite terms of positive polarity, in the opposite direction to the implication, and terms of negative polarity in the direction of the implication. These polarity consideration are all dealt with by the rewriting mechanism in $\lambda Clam$.

$\lambda Clam$ also has the capability of reasoning using *conditional* rewrite rules. This means that whenever a condition is attached to the rewrite rule, it must be satisfied in order for the rule to be applicable. Consider the field axiom

$$\forall x : \tau .\ x \neq 0 \rightarrow x \times x^{-1} = 1.$$

We can use this axiom as a rewrite rule in a number of different ways:

$$\begin{array}{lllll} X \neq 0 & \rightarrow & X \times X^{-1} & \Rightarrow & 1 \\ X \neq 0 & \rightarrow & 1 & \Rightarrow & X \times X^{-1} \\ & & X \times X^{-1} = 1 & \Rightarrow & X \neq 0. \end{array}$$

According to the usual notion of rewriting, the second of these rules causes the set to be non-confluent and non-terminating. In $\lambda Clam$ it is possible to apply such a rule once, but its application must be controlled. When we apply these rewrite rules to the conclusion of a goal, we

observe different behaviours. If the first is applied then the subgoal $X \neq 0$ must be established for the instantiation of X which is found. If the second is applied then the subgoal $X \neq 0$ must be set, but the subgoal will not be provable until an instantiation for X is found by proceeding with the proof of the goal. If the third is applied then the entire formula $X \times X^{-1} = 1$ is rewritten and the formula $X \neq 0$ is replaced.

In subsequent chapters we give examples of how λ*Clam* automatically speculates lemmas if rewriting or rippling cannot apply to a goal. We show how the lemma is stated, and a form chosen for the representation as a rewrite rule.

Since λ*Clam* is a higher-order proof-planner, we can use higher order rewrite rules. This is important to this research because we find proof-plans for some higher-order theorems. We shall see an example of higher-order axioms in our axiomatisation for non-standard analysis given in section 4.2.4.

4.2.4 The axiomatisation

In this section, we present the axioms of our system, by considering the different notions involved in our work. For each of these, we first introduce the constant symbols and their types, and then the axioms with which we augment the sequent calculus described in section 4.2.1. It is possible in λ*Clam* to overload function symbols, by giving them more than one type, and this is shown here. We present these axioms as they are represented in λ*Clam* . For example where we write $0 : \mathbb{R}, {}^*\mathbb{R}$, internally 0 is typed both as a real number and as a hyperreal. This means that when an axiom is unquantified, it is assumed to be universally quantified, and is represented internally as an unquantified rewrite rule. Where quantification is explicit in rewrite rules, we use the notation $\forall x : \mathbb{R}$ to distinguish from the notation for typing in goals, where we write $\forall x \in \mathbb{R}$, as can be seen in the rules for quantifiers in the sequent calculus shown in table 4.1.

Peano axioms for arithmetic

We introduce the following constants here:

$$\begin{aligned} 0 :&\quad \mathbb{N},\ {}^*\mathbb{N} \\ s :&\quad \mathbb{N} \to \mathbb{N},\ {}^*\mathbb{N} \to {}^*\mathbb{N} \\ + :&\quad \mathbb{N} \times \mathbb{N} \to \mathbb{N},\ {}^*\mathbb{N} \times {}^*\mathbb{N} \to {}^*\mathbb{N} \\ \times :&\quad \mathbb{N} \times \mathbb{N} \to \mathbb{N},\ {}^*\mathbb{N} \times {}^*\mathbb{N} \to {}^*\mathbb{N} \end{aligned}$$

$$.^x : \quad \mathbb{N} \times \mathbb{N} \to \mathbb{N},\ {}^*\mathbb{N} \times {}^*\mathbb{N} \to {}^*\mathbb{N}$$

Notice here that we only consider axioms for functions of one argument. In order to generalise the notion of extension to functions of more than one argument requires more work than was feasible during this work. If we were able to generalise the notion of extension to functions with an arbitrary number of arguments would extend the number of theorems possible for consideration, and allow us to define extensions for functions of arity two, such as $+$, which would in turn would allow us to generate field axioms for the hyperreals from those from the reals. The drawback of generalising the notion of functional extensions to functions of an arbitrary number of arguments is that we would complicate our represenation. We would have to introduce an inductive datatype such as lists in order to represent the arguments to a function. The Peano axioms are:

$$0 \in \mathbb{N} \tag{4.1}$$

$$\forall x.\ s(x) \neq 0 \tag{4.2}$$

$$\forall x, y.\ s(x) = s(y) \to x = y \tag{4.3}$$

We introduce induction as an inference rule, only over the natural numbers:

$$\frac{\Gamma \vdash P(0) \quad \Gamma, y : \mathbb{N}, P(y) \vdash P(s(y))}{\Gamma, x : \mathbb{N} \ \vdash \ P(x)}\ \texttt{ind}$$

This is true over the hypernaturals if perceived as an axiom schema where P can be instantiated to any first order predicate expressible in the language of Peano arithmetic. Equivalently stated, this holds as long as P is internal in the non-standard model. For a discussion of this see section 4.3.

We also add the following definitions for $+$, $\times$ and exponentiation:

$$0 + x = x \tag{4.4}$$

$$0 \times x = 0 \tag{4.5}$$

$$x^0 = s(0) \tag{4.6}$$

$$s(x) + y = s(x + y) \tag{4.7}$$

$$s(x) \times y = (x \times y) + y \tag{4.8}$$

$$x^{s(y)} = x \times x^y \tag{4.9}$$

We define the reflexivity of equality:

$$A = A \tag{4.10}$$

This is represented internally as a rewrite rule, and not as an inference rule, as discussed in section 4.2.3.

The field axioms

We introduce the field axioms in this section. A standard axiomatisation can be found in [Apostol, 1974]. Wh en axioms are stated without explicit quantification, it is assumed that the variables range over both the reals and the hyperreals.

Firstly we introduce the following constants:

$$\begin{aligned}
0: &\quad \mathbb{R}, {}^*\mathbb{R} \\
1: &\quad \mathbb{R}, {}^*\mathbb{R} \\
+: &\quad \mathbb{R}\times\mathbb{R}\to\mathbb{R},\ {}^*\mathbb{R}\times{}^*\mathbb{R}\to{}^*\mathbb{R} \\
\times: &\quad \mathbb{R}\times\mathbb{R}\to\mathbb{R},\ {}^*\mathbb{R}\times{}^*\mathbb{R}\to{}^*\mathbb{R} \\
-: &\quad \mathbb{R}\to\mathbb{R},\ {}^*\mathbb{R}\to{}^*\mathbb{R} \\
.^{-1}: &\quad \mathbb{R}\to\mathbb{R},\ {}^*\mathbb{R}\to{}^*\mathbb{R} \\
<: &\quad \mathbb{R}\to\mathbb{R}\to\mathbb{B},\ {}^*\mathbb{R}\to{}^*\mathbb{R}\to\mathbb{B}
\end{aligned}$$

As discussed in section 2.2.1, the following theorem expresses the completeness of the reals:

$$\begin{aligned}
\forall P\in\mathbb{R}\to\mathbb{B}.\ (\exists x\in\mathbb{R}.\ P(x))\wedge(\exists U\in\mathbb{R}.\ \forall y\in\mathbb{R}.\ P(y)\to y\leq U)\to \\
\exists u\in\mathbb{R}.\ (\forall x\in\mathbb{R}.\ P(x)\to x\leq u)\wedge \\
\forall u'\in\mathbb{R}.\ (\forall x\in\mathbb{R}.\ P(x)\to x\leq u')\to\ u\leq u'
\end{aligned}\tag{4.11}$$

We do not include this as an axiom in our system because it is not required explicitly for any of our proof-plans. It must be noted at this stage that the fact that we do not use the completeness of the reals explicitly in our axiomatisation is because completeness is implicitly contained within some of the axioms (axiom (4.40)) for example. Specifically, in order to prove these axioms in Isabelle/HOL, the completeness of the reals must be used. As will be seen in Chapter 6, hyperreals can be defined using infinite hypernaturals whereby some uses of completeness can be obviated.

The ordered field axioms are:

$$1 \neq 0 \quad (4.12)$$
$$x+(y+z) = (x+y)+z \quad (4.13)$$
$$(-x)+x = 0 \quad (4.14)$$
$$x \times (y \times z) = (x \times y) \times z \quad (4.15)$$
$$x \neq 0 \to x \times x^{-1} = 1 \quad (4.16)$$
$$x \times (y+z) = (x \times y) + (x \times z) \quad (4.17)$$
$$x < y \wedge y < z \to x < z \quad (4.18)$$
$$y < z \to x+y < x+z \quad (4.19)$$
$$x+y = y+x \quad (4.20)$$
$$0+x = x \quad (4.21)$$
$$0 \times x = 0 \quad (4.22)$$
$$x \times y = y \times x \quad (4.23)$$
$$1 \times x = x \quad (4.24)$$
$$x = y \vee x < y \vee y < x \quad (4.25)$$
$$x \not< x \quad (4.26)$$
$$0 < x \wedge 0 < y \to 0 < x \times y \quad (4.27)$$

There are two important properties of a real closed field, which we do not include in our axiomatisation as they are not necessary in any of the proofs we study. For an ordered field $\mathcal{K}$ these are:

1. Every positive element of $\mathcal{K}$ has a square root in $\mathcal{K}$;
2. Every polynomial $f(x) \in \mathcal{K}$ of odd degree has a root in $\mathcal{K}$

Extra axioms for non-standard analysis

We introduce the following constant symbols:

$$\begin{aligned} \mathit{emb}: &\quad \mathbb{N} \to {}^*\mathbb{N},\ \mathbb{R} \to {}^*\mathbb{R} \\ \mathit{finite}: &\quad {}^*\mathbb{R} \to \mathbb{B},\ {}^*\mathbb{N} \to \mathbb{B} \\ \approx: &\quad {}^*\mathbb{R} \to {}^*\mathbb{R} \to \mathbb{B} \\ \mathit{ext}: &\quad (\mathbb{R} \to \mathbb{R}) \to ({}^*\mathbb{R} \to {}^*\mathbb{R}), \\ &\quad (\mathbb{N} \to \mathbb{N}) \to ({}^*\mathbb{N} \to {}^*\mathbb{N}), \\ &\quad (\mathbb{N} \to \mathbb{R}) \to ({}^*\mathbb{N} \to {}^*\mathbb{R}) \end{aligned}$$

We treat $\mathbb{N}$ as a subtype of $\mathbb{R}$, leaving the relation between the two implicit in the axiomatisation. We define $\approx$ as an equivalence relation:

$$A \approx A \quad (4.28)$$
$$A \approx B \to B \approx A \quad (4.29)$$
$$A \approx B \ \wedge \ B \approx C \to A \approx C \quad (4.30)$$

We need to distinguish between equality and the infinitely close relation over the hyperreals:

$$\forall A, B : {}^*\mathbb{R}.\ A = B \rightarrow A \approx B \tag{4.31}$$

$$\forall X : {}^*\mathbb{R}.\ \exists Y : {}^*\mathbb{R}.\ X \approx Y \wedge X \neq Y \tag{4.32}$$

We also to make a distinction between the naturals and the hypernaturals:

$$\exists M : {}^*\mathbb{N}.\ \forall N : \mathbb{N}.\ M > \widehat{N} \tag{4.33}$$

We state the closure axioms for the embedding operator `emb`:

$$\forall X, Y : \mathbb{R}.\ \ \widehat{X} + \widehat{Y} = \widehat{X + Y} \tag{4.34}$$

$$\forall X, Y : \mathbb{R}.\ \ \widehat{X} \times \widehat{Y} = \widehat{X \times Y} \tag{4.35}$$

$$\forall X : \mathbb{R}.\ \ X \neq 0 \ \rightarrow \qquad \widehat{X}^{-1} = \widehat{X^{-1}} \tag{4.36}$$

$$\forall X : \mathbb{R}.\ \ -\widehat{X} = \widehat{-X} \tag{4.37}$$

We must add some axioms which distinguish the reals from the hyperreals, and the finite numbers from the reals:

$$\forall X : {}^*\mathbb{R}. \forall Y : \mathbb{R}.\ \ X \approx \widehat{Y} \rightarrow \mathit{finite}(X) \tag{4.38}$$

$$\forall X : {}^*\mathbb{R}.\ \mathit{finite}(X) \rightarrow \exists Y : \mathbb{R}.\ X \approx \widehat{Y} \tag{4.39}$$

$$\forall X, Y : \mathbb{R}.\ \widehat{X} \approx \widehat{Y} \rightarrow X = Y \tag{4.40}$$

$$\exists X : {}^*\mathbb{R}.\ \forall Y : \mathbb{R}.\ \ \mathit{finite}(X) \wedge X \neq \widehat{Y} \tag{4.41}$$

$$\forall X, Y : {}^*\mathbb{R}.\ \ \mathit{finite}(X) \wedge \mathit{finite}(Y) \rightarrow \mathit{finite}(X + Y) \tag{4.42}$$

$$\forall X, Y : {}^*\mathbb{R}.\ \ \mathit{finite}(X) \wedge \mathit{finite}(Y) \rightarrow \mathit{finite}(X \times Y) \tag{4.43}$$

$$\forall X : {}^*\mathbb{R}.\ \ \mathit{finite}(X) \rightarrow \mathit{finite}(-X) \tag{4.44}$$

$$\forall X : {}^*\mathbb{R}.\ \ X \not\approx 0 \rightarrow \mathit{finite}(X^{-1}) \tag{4.45}$$

We state the closure axioms for non-standard extensions:

$$\forall f, g : \mathbb{R} \rightarrow \mathbb{R}.\ \forall x : {}^*\mathbb{R}.\ \ {}^*f(x) + {}^*g(x) \ = \ {}^*(\lambda y.(f(y) + g(y)))x \tag{4.46}$$

$$\forall f, g : \mathbb{R} \rightarrow \mathbb{R}.\ \forall x : {}^*\mathbb{R}.\ \ {}^*f(x) \times {}^*g(x) \ = \ {}^*(\lambda y.(f(y) \times g(y)))x \tag{4.47}$$

$$\forall f : \mathbb{R} \rightarrow \mathbb{R}.\ \forall x : {}^*\mathbb{R}.\ \ -{}^*f(x) \ = \ {}^*(\lambda y. -f(y))x \tag{4.48}$$

$$\forall f : \mathbb{R} \rightarrow \mathbb{R}.\ \forall x : {}^*\mathbb{R}.\ \ {}^*f(x)^{-1} \ = \ {}^*((\lambda y. f(y)^{-1}))x \tag{4.49}$$

$$\forall f, g : \mathbb{R} \rightarrow \mathbb{R}.\ \forall x : {}^*\mathbb{R}.\ \ \lambda x.\, {}^*f({}^*g(x)) \ = \ {}^*(\lambda y.\ f(g(y))) \tag{4.50}$$

We must distinguish functions in the hyperreals domain from standard functions:

$$\exists f : {}^*\mathbb{R} \to {}^*\mathbb{R}.\ \forall g : \mathbb{R} \to \mathbb{R}.\quad f \neq {}^*g \tag{4.51}$$

$$\forall X, Y : \mathbb{R}.\ \forall f : \mathbb{R} \to \mathbb{R}.\quad f(X) = Y \iff {}^*f(\widehat{X}) = \widehat{Y} \tag{4.52}$$

We formalise the behaviour of the field operators around an infinitesimal neighbourhood:

$$\forall X, Y, Z : {}^*\mathbb{R}.\quad X \approx Z \wedge finite(Y) \to X \times Y \approx Z \times Y \tag{4.53}$$

$$\forall X, Y : {}^*\mathbb{R}.\quad X \approx Z \wedge Y \approx 0 \to X + Y \approx Z \tag{4.54}$$

$$\forall X : {}^*\mathbb{R}.\quad X \approx 0 \to -X \approx 0 \tag{4.55}$$

$$\forall X : {}^*\mathbb{R}.\quad X \approx 0 \to \neg\, finite(X^{-1}) \tag{4.56}$$

We add an inference rule which describes transfer of a goal from the standard domain to the non-standard domain (using the definition of the *-transform given in section 2.4.1):

$$\frac{\Gamma \vdash \phi}{{}^*\Gamma \vdash {}^*\phi} \tag{4.57}$$

Notes on the axiomatisation

We claim that the logic given in section 4.2.1 augmented with the axioms given in is a sound axiomatisation of non-standard analysis, but we do not claim completeness. We cannot guarantee that there may be some true statement in the non-standard domain that we are unable to prove using this axiomatisation, since we have arithmetic for the natural numbers. The chosen axioms are theorems of the Isabelle/HOL theorem prover, thanks to the construction of the hyperreals performed by Fleuriot therein [Fleuriot, 2001a].

The axiomatisation outlined is a presentation of the rules which make up the non-standard analysis theory of $\lambda Clam$. In many cases the application of rules is controlled. It is also important to note that the atomic methods which are presented in subsequent chapters often correspond to the application of many of these axioms. We justify their behaviour in terms of these axioms, but as we do not claim to yield object-level proofs, our proof-plans do not correspond to sequences of applications of these axioms. It will be made clear how such methods are developed, and why this is the case, in subsequent chapters.

Although the axioms presented in this chapter are those we have implemented in the $\lambda Clam$ system, there are various interdependencies which should be noted. Firstly, we can derive axiom (4.41) from axioms (4.38) and (4.32), by simply choosing a real Y in axiom (4.32). Also

axiom (4.51) can be derived from axioms (4.52) and (4.41). Also an equivalent and perhaps neater version of (4.52) would be

$$\forall X : \mathbb{R}.\ \forall f : \mathbb{R} \to \mathbb{R}.\ \ {}^*f(\widehat{X}) = \widehat{f(X)}.$$

The axioms we have presented here represent those implemented in $\lambda Clam$. No claim is made about these forming a minimal set.

4.3 On the transfer principle

This section discusses how simpler definitions and proofs can be achieved, and explains how there is a choice to made in the representation. The discussion that follows is relevant to this research, because we make choices about representation throughout. In order to show that we are reasoning about the same problems as are stated in real analysis, we must justify this choice.

4.3.1 Use of the transfer principle

We use the transfer principle directly in our work, by employing an instance of the meta-theorem in both directions to validate statements in both the standard and the non-standard domain. We use rule (4.40) as an implementation of transfer from the non-standard to the standard domain:

$$\forall X, Y : \mathbb{R}.\ \widehat{X} \approx \widehat{Y} \to X = Y.$$

This incorporates the rule

$$\forall X, Y : \mathbb{R}.\ \widehat{X} \approx \widehat{Y} \to \widehat{X} = \widehat{Y}$$

and an implementation of transfer from the hyperreals to the reals.

When we deduce results in the standard domain and transfer them to the non-standard domain, we use instances of inference rule (4.57) representing the transfer meta-theorem. An example of such an instance of this rule is given in section 6.5.5 of chapter 6. We do not explicitly implement the transfer principle in our work, instead preferring to use instantiations of it where necessary. In order to correctly formalise the transfer principle a function which is recursive on the term structure is necessary.

As an example of a function carrying out transfer, consider the following function `termrec`, which is recursively defined over the term structure. We restrict this exposition to functions of a single argument for simplicity. We use $\to$ to represent the conditions under which a rule

applies.

$$\texttt{termrec}\ \ \textit{quant\_var} \quad = \quad \textit{quant\_var}$$

$$x \neq \textit{quant\_var} \quad \rightarrow \quad \texttt{termrec}\ \ x \quad = \quad \texttt{termrec}\ \ \widehat{x}$$

$$\texttt{termrec}\ \ f(x) \quad = \quad {}^{*}f(\texttt{termrec}\ \ x)$$

$$\texttt{termrec}\ \ \forall x : \tau.\ (\lambda y.P(y))\ \ x \quad = \forall x' : {}^{*}\tau.\ \ (\lambda\ \ \textit{quant\_var.}\ \ \texttt{termrec}\ \ P(\textit{quant\_var}))\ \ x'$$

The behaviour of this function formalises the transfer principle as described in section 2.4.1. A precise definition of transfer, and the conditions under which it is applicable can be seen in [Robinson, 1966].

4.3.2 Limits and continuity

As shown in chapter 2, the definitions of limit and continuity are very much simplified in non-standard analysis. In order to justify its use, it is important that this simplified definition be explained. The transfer theorem, as stated in 2.4.1, allows us to transfer sentences expressed in standard terms to the non-standard domain.

Recall definitions 2, and 9 from chapter 2. The proof of the equivalence involves reasoning "across" the models. What is in fact happening in the proof is that there is reasoning being done about both the *internal* sentence of the $*$-transformed standard definition, and the *standard* sentence of the standard definition expressed entirely in the non-standard model. It is important to note that in what follows we are presenting a pen and paper proof. We do not claim to have planned this proof in $\lambda Clam$. Let us just prove that the standard definition (2) implies the non-standard definition (9):

Suppose first that $\lim_{x\to a}\ f(x) = l$. Now fix a hyperreal, α, such that $\alpha \approx \widehat{a}$ and $\alpha \neq \widehat{a}$. We need to show that ${}^{*}f(\alpha) \approx \widehat{l}$. We know from the standard definition that:

$$\forall \varepsilon \in \mathbb{R}.\ \varepsilon > 0 \rightarrow \exists \delta \in \mathbb{R}.\ \delta > 0 \ \wedge\ \forall x \in \mathbb{R}. \quad 0 < |x-a| < \delta \rightarrow |f(x) - l| < \varepsilon$$

Now let n be a standard natural number, and let $\varepsilon = \frac{1}{n}$. By definition, we thus have a δ for which:

$$\forall x \in \mathbb{R}. \quad 0 < |x-a| < \delta \rightarrow |f(x) - l| < \frac{1}{n}$$

now by transfer we have

$$\forall x \in {}^{*}\mathbb{R}. \quad 0 < |x-\widehat{a}| < \widehat{\delta} \rightarrow |{}^{*}f(x) - \widehat{l}| < \frac{1}{\widehat{n}}$$

Now let $x = \alpha$. As $\alpha - a \approx 0$, $|x - \alpha|$ must be less than all positive real numbers, so we have

$$|^* f(\alpha) - \widehat{l}| < \frac{1}{\widehat{n}}$$

Since n is standard and arbitrary, this means that $|^* f(\alpha) - \widehat{l}|$ is less than all positive real numbers (this follows directly from the Archimedean property of the reals), and hence $^* f(\alpha) \approx \widehat{l}$ as required.

This proof is interesting because the choice at which transfer was applied was vital to the correct conclusion to the proof. The reason why this proof succeeds is that n has to be a standard natural number at the end, as does δ, so these quantifiers were eliminated. If transfer had been applied earlier we would not have been able to introduce the $\approx$ relation. If transfer had been applied after x had been fixed as a real, we could not have used α to yield the result.

4.3.3 Induction

The Peano axioms for the natural numbers can be transferred to the hypernatural domain, meaning that induction is possible on the hypernaturals. Any instantiated induction scheme can also be transferred as it constitutes a first order standard sentence. An example of an uninstantiated induction scheme is:

$$\frac{\Gamma \vdash P(0) \quad \Gamma, y : \mathbb{N}, P(y) \vdash P(s(y))}{\Gamma, x : \mathbb{N} \vdash P(x)} \ \texttt{ind}$$

which can be written as follows in the non-standard domain:

$$\frac{\Gamma \vdash {}^*P(0) \quad \Gamma, y : {}^*\mathbb{N}, {}^*P(y) \vdash {}^*P(s(y))}{\Gamma, x : {}^*\mathbb{N} \vdash {}^*P(x)} \ \texttt{ind}.$$

This has a higher-order variable P that can be instantiated and then the scheme transferred to the hyperreal domain, where x represents the variable chosen for induction. For example the theorem

$$\forall x, y \in \mathbb{N}.\ x + s(y) = s(x + y)$$

has an instantiated induction scheme:

$$\frac{\vdash \forall y \in \mathbb{N}.\ 0 + s(y) = s(0 + y) \quad x : \mathbb{N}, \forall y \in \mathbb{N}.\ x + s(y) = s(x + y) \vdash \forall y \in \mathbb{N}.\ s(x) + s(y) = s(s(x) + y)}{z : \mathbb{N} \vdash \forall y \in \mathbb{N}.\ z + s(y) = s(z + y)} \ \texttt{ind}$$

This could be expressed in non-standard analysis by transferring it to the hypernaturals. Equally the induction could be performed entirely over the standard naturals and the resulting theorem transferred to the hypernatural domain. This research adopts the latter approach, planning proofs of conjectures in the standard domain, and then transferring the results across

to non-standard analysis. We choose this approach because we can then draw on the considerable work already done in $\lambda Clam$ on developing plan specifications for inductive proofs.

4.3.4 Choice of representation

When standard theorems are presented in this work, we must decide how to state the hypotheses and conclusions of a goal in non-standard analysis. As can be seen from the proof in section 4.3.2 we must choose at which stage in the proof to transfer statements. In some cases a wrong choice will lead to a harder proof when reasoning entirely in the non-standard model.

In order to exemplify this, let us consider the statement of the Intermediate Value Theorem:

Theorem 6 *Let f be a function, $f : \mathbb{R} \to \mathbb{R}$, which is continuous on the closed interval [a, b]. Suppose that c is a real number between f(a) and f(b); then there exists x in [a, b] such that f(x) = c.*

We must consider how to state this theorem in non-standard analysis. In some text books, the c here is presented as a universally quantified variable in the conclusion, and in some it is a parameter. In the standard case these two steps are equivalent, but the transfer principle means the non-standard versions of these two statements are different.

In the first approach, where the standard conclusion is:

$$\forall c \in \mathbb{R}.\ f(a) \leq c \leq f(b) \rightarrow\ \exists x \in \mathbb{R}\ f(x) = c$$

the $*$-transformed conclusion is:

$$\forall c \in {}^*\mathbb{R}.\ \widehat{f(a)} \leq c \leq \widehat{f(b)} \rightarrow\ \exists x \in {}^*\mathbb{R}\ {}^*f(x) = c$$

and in the second approach, where the standard conclusion is:

$$f(a) \leq c \leq f(b) \rightarrow\ \exists x \in \mathbb{R}\ f(x) = c$$

the $*$-transformed conclusion is:

$$\widehat{f(a)} \leq \widehat{c} \leq \widehat{f(b)} \rightarrow\ \exists x \in {}^*\mathbb{R}\ {}^*f(x) = \widehat{c}$$

where c is now of type `real`.

In the current work, as can be seen in section 6.2 we opt for the second approach when specifying non-standard versions of familiar theorems.

4.4 Evaluation Methodology

Each plan-specification that is introduced into the system must be evaluated according to various criteria. The performance of the `ripple_out` method, which is used in most of the inductive proofs here, is evaluated according to many criteria in [Bundy, 1989]. Some of these criteria apply to plan-specifications, and some to the resulting proof-plans. In our research we are interested in judgements about the plan-specifications, but we also make references to the criteria pertaining to the proof-plans themselves.

4.4.1 Possible evaluation criteria

The following criteria describe some of the ways in which plan specifications can be evaluated. This follows closely the discussion given in [Bundy, 1989]. In the current discussion however, we split the criteria up into two categories: the output criteria pertaining to the *proof-plan* which analyse what the planner produces, the criteria about the *plan-specification*, which measure the suitability of the plan-specification, and the *process* criteria which analyse how the planner goes about producing its results. This categorisation of criteria has not been done previously.

Proof-plan criteria

- **Expectancy**
  A proof-plan has *expectancy* if we have a means of predicting its success. A formal way of stating this is that if the preconditions to the proof plan are met by the input to the tactic then the output to the tactic will meet the effects of the proof-plan.

- **Correctness**
  A proof-plan is said to be *correct* if its execution at the object level produces a proof.

Plan-specification criteria

- **Generality**
  A plan-specification is said to be more *general*, the more theorems it produces complete proof-plans for. Generality is a measure that can be applied to all plan-specifications, but cannot be judged alone, or there will be a tendency to build over-complicated plan-specifications. For more discussion on this see section 4.4.2.

- **Intuitiveness**
  A plan-specification is said to be *intuitive* if the proof-plans it produces correspond to the structure of the proof when performed by a human. It is difficult to quantify the intuitiveness of a single proof-plan, but we gain some idea by measuring how many steps correspond to the human pattern of reasoning. We can then produce an overall measure of intuitiveness by assessing all of the proof-plans that the plan-specification creates.

- **Simplicity**
  A plan-specification is given more credit the more concisely it is stated. The resulting proof-plan may be very complex. However, *simplicity* applies to plan-specifications.

Process criteria

- **Prescriptiveness**
  A plan-specification is said to be *prescriptive* if it performs little search in finding a proof-plan.

- **Efficiency**
  A plan-specification is given more credit if it is *efficient*. This means that the process of finding a complete proof-plan, given the plan-specification, is not computationally expensive.

4.4.2 Discussion of evaluation issues

We discuss some of the issues which are relevant when constructing a scheme for evaluation. We justify the choice of evaluation measurements given in section 4.4.3, by explaining some of the factors that affect the success of proof-planning.

The set of plan-specifications

For any particular theory we would ideally attain, through proof-planning, one succinct plan-specification which would account for all of the theorems in the development set, but this is rarely the case. It is possible of course to make one large proof plan in order to account for all of the development examples, by making a large over complicated plan-specification which incorporates the others. This approach would score badly on *prescriptiveness*, *simplicity* and

efficiency. We would prefer in this case to have a small set of prescriptive and simple proof-plans, which are fundamentally different, to account for the theory.

Evaluation in the presence of Critics

As described in [Ireland, 1992], critics are a proof-planning tool, which react to the failure of methods by analysing the failure of the preconditions to apply. A plan-specification can be very simple in its structure of method applications, but there may be very many critics attached to it, which themselves perform complex proof transformations.

It may be the case that one plan-specification may account for an entire theory, and so should be rated highly in terms of *generality*, but it is not clear how well it would fare according to the other criteria. Our attitude is to favour a plan-specification which is simple, but has many critics attached to it. If the general form of the majority of proofs in a certain class can be found and captured in a general plan-specification, which is transformed by the application of critics, then this is preferable, as we claim that such a plan-specification is more *intuitive*, *general* and *prescriptive*. This approach will certainly be less *efficient* than having a set of plan-specifications, but we believe that this is a reasonable price to pay. The object of finding plan-specifications in this research is to understand the structure of our mathematical theory.

In order to argue for *generality*, *simplicity* and *efficiency*, simple quantitative measures can be made to determine whether the plan-specifications work better with respect to a certain criterion, with or without critics attached.

In order to argue for a plan-specification being more *intuitive* with critics attached, we need to argue that our reasoning follows the same pattern during the proofs. Critics are an attractive part of the proof-planning paradigm because they help mimic the reasoning performed by a human during the proof process. It is not the case that when a branch is searched that we forget all information yielded when we find the way is blocked. In general we analyse the failure of a proof and alter it according to the knowledge gained by the failure. For this reason we argue that simple plan-specifications with critics attached have a more intuitive structure than those which are comprised entirely of methods.

In order to argue that a plan-specification is more *prescriptive* we can appeal to the fact that we put stringent preconditions on methods which reduce the search space. This can be measured for example by investigating the factors affecting the size of the search tree, such as the branching factor at each node.

We claim that the methodology of constructing simple but general plan-specifications with

associated sets of critics, is a more intuitive way of performing proof. It reproduces the behaviour of proofs performed by humans more closely than a larger set of more specific plan-specifications. We give our set of experiments and evaluation scheme in section 4.4.3

Evaluation with intermediate lemmas

One of the criteria by which we judge the success of proof-plans is how many lemmas were needed. In previous work [Bundy, 1988] this was very relevant because the lemmas were added to the system as rewrite rules by the user. This will be the case for some of the theorems which are planned in this research also.

It is a goal of this work to obviate the need to add by hand anything that does not appear in our axiomatisation. We plan to incorporate lemma speculation critics, as first devised in [Ireland and Bundy, 1996], to guess the rewrite rule needed to complete the proof at any time, and to to set it as a subgoal. Very often these speculated rules are equational and can easily be verified by a computer algebra system. The mathweb system [Franke and Kohlhase, 1999, Franke et al., 1999] can be used to send these subgoals out to such programs. In this case the plan-specification should not be penalised in its evaluation, because the lemmas were speculated and verified by the planner during the proof, hence there is no loss of *generality*.

User supplied definitions

As described in chapter 6, there is a class of theorems whose proofs involve induction. These proofs have a modular form: they are comprised of several inductive lemmas, and of a final stage of the proof which uses these lemmas to deduce the required result. The inductive lemmas relate to a recursive definition supplied by the user.

As it is not the case that the original conjecture is given on its own to the planner, a measurement has to be made here about how “automatic” these proofs are. One simple measure that can be made is to count how many recursive functions and lemmas had to be stated by hand in order for the final part of the proof, which uses these lemmas, to be planned automatically. Also, these proofs may individually require separate plan-specifications; the intermediate lemmas result in inductive proof-plans, but the final part of the proof will need a plan-specification for reasoning in non-standard analysis.

4.4.3 Our evaluation scheme

For our evaluation we intend to judge our proof-plans only by the *specification* criteria, and the *process* criteria. As mentioned in section 4.1.3, we do not claim to execute the proof-plans in an object level theorem prover, and are hence less interested in the *output* criteria.

For each theorem and for each plan-specification, we make the following measurements.

1. Was a complete proof-plan yielded?
2. Number of lemmas automatically speculated
3. Number of user defined lemmas
4. Size of eventual proof-plan
5. Size of successful plan-specification
6. Average branching factor per node
7. Number of critics fired

We can make immediate judgements about the *generality*, *efficiency*, *simplicity* and *prescriptiveness* of the plan-specifications.

In order to make a judgement about whether the plan-specification are *intuitive*, we should perform experiments to map the proof-plans to actual human proof. One simple way of showing that the plan specifications are not intuitive is to show that the resulting proof-plans are not understandable. If they are not understandable, then the plan-specification can surely not be intuitive. It may be possible to investigate human proof using these tools, by introducing an interactive component to the system at important choice-points.

4.5 Summary

Up to this point we have set out an axiomatisation, and a proof-planning framework, by which we can construct plan specifications for proofs using non-standard analysis. We have also set out a methodology for evaluating the success of the research.

We go on to use this methodology to understand the structure of certain classes of proof in non-standard analysis. The combination of proof-planning and non-standard analysis has not previously been investigated.

Chapter 5

Proof-planning limit theorems

This chapter describes the work that was done to implement the automation of plan construction for conjectures involving limits and continuity. We first describe the implementational research contribution of the work presented in this chapter. We go on to present proofs-plans for the theorems in our development set. The proof shown for each theorem matches exactly the steps taken by the proof-plan. We then describe the reasoning observed in these proofs, and describe the specifications of methods which encapsulate some of these patterns. We show the critics we construct, and the set of plan-specifications we define to account for the majority of the development set. We finally describe the application of the resulting planning machinery to a new set of theorems- the test set- and analyse the results.

5.1 System enhancement

Apart from the plan-specifications and proof-plans we yield for these proofs, we have also provided $\lambda Clam$ with some necessary functionality which is now in use.

Coloured Rippling

We have altered the rippling methods so that it is possible to embed more than one hypothesis in the conclusion of a goal sequent, and thence follow the progress of the wave fronts. This is a generalisation of rippling with just one hypothesis, as is usually done in inductive proofs, and does not affect existing theorems since coloured rippling generalises to reasoning with one or more hypotheses.

Lemma speculation

We have implemented a mechanism to allow speculated lemmas to be used as rewrite

rules during the execution of a proof-plan. Previously this was not possible, and lemmas had to be introduced by adding them to the hypotheses of a sequent. This is not desirable, as often conditional rewrite rules need to be specified where the condition is separated from the left and right side. Our mechanism allows a lemma to be added a rewrite-rule to the context, so that it can be used to rewrite the goal.

Conditional Rewriting

Another significant implementational device we have introduced is the ability to reason using conditional rewrite rules. Before, it was assumed that when rewriting with conditional rewrite rules the condition had to be part of the hypotheses. Now we treat the condition as a separate goal, and try to establish a plan for the proof of the condition first, thus justifying its use in the original goal.

5.2 Reasoning patterns from the proofs

We present here the theorems which constitute the development set for our system. We constructed a specific proof-plan for each theorem, and the proofs described here follows the steps dictated by the proof-plan. We also discuss the patterns of reasoning we find in the set of proofs. Importantly we use non-standard characterisation of uniform continuity in the theorems we study here, but we do not use uniform differentiability, as it is possible to yield proof-plans using the normal non-standard characterisation shown in theorem 5 of section 2.4.4.

5.2.1 Development examples

Our methodology for constructing general plan-specifications is to generate proof-plans in $\lambda Clam$ for the development set, and to analyse them to determine how best to generalise the reasoning patterns. When we use rewrite rules in the proof-plans, we must verify that they are valid using our axiomatisation, by stating lemmas which justify the use of each rule. This increases the size of the development set significantly. Lemmas are often needed in order to complete the proofs, and we present proofs of the non-trivial ones here. Where possible we give the theorems and lemmas names, but in some cases we refer to them by number. We highlight some reasoning patterns in the presentations of the proof-plans by mentioning them in bold font. These are explained in more detail in section 5.2.2.

LIM+

Let us take as a first motivating example the conjecture $LIM+$ which states that the sum of limits of two functions at a particular point is equal to the limit of the sums at the point. We write this is as

$$\lim_{x\to c} f(x) = l_f \wedge \lim_{x\to c} g(x) = l_g \vdash \lim_{x\to c} f(x)+g(x) = l_f + l_g.$$

When the theorem is written out formally using the non-standard characterisation of a limit, we yield the conjecture

$$c, l_f, l_g : \mathbb{R} \quad f, g : \mathbb{R} \to \mathbb{R}$$
$$(\forall x \in {}^*\mathbb{R}.\ x \approx \widehat{c} \wedge x \neq \widehat{c} \to {}^*f(x) \approx \widehat{l_f}) \wedge (\forall x \in {}^*\mathbb{R}.\ x \approx \widehat{c} \wedge x \neq \widehat{c} \to {}^*g(x) \approx \widehat{l_g}) \vdash$$
$$\forall x \in {}^*\mathbb{R}.\ x \approx \widehat{c} \wedge x \neq \widehat{c} \to {}^*(\lambda z.\, f(z)+g(z))(x) \approx \widehat{l_f + l_g}.$$

Before performing any proof steps we can notice important facts about the shape of the conjecture. Firstly, in the form stated all the hypotheses that are not simply typing information embed into the conclusion so that it is possible to use annotated reasoning techniques, such as those described in section 3.2.5. After the application of rule $r\forall$, we embed the hypotheses in the conclusion, and after the application of rules (4.46) and (4.34) we see that the conclusion becomes

$$x \approx \widehat{c} \wedge x \neq \widehat{c} \to \boxed{{}^*f(x) + {}^*g(x)}^{\uparrow} \approx \boxed{\widehat{l_f} + \widehat{l_g}}^{\uparrow}.$$

Now we want to rewrite the conclusion so that the terms in like-coloured wave holes move together after rewriting. Eventually we want each wave hole to correspond precisely to an instantiation of a hypothesis. In this case we need to use the two wave rules

$$\boxed{X + A}^{\uparrow} \approx \boxed{Y + B}^{\uparrow} \Rightarrow \boxed{X \approx Y \wedge A \approx B}^{\uparrow} \tag{5.1}$$

$$A \to \boxed{B \wedge C}^{\uparrow} \Rightarrow \boxed{A \to B \wedge A \to C}^{\uparrow} \tag{5.2}$$

and rules (4.46) and (4.34) which can also be annotated. Embedding the hypotheses in the conclusion and rewriting using these wave rules is an example of **rippling out**, which is a reasoning pattern we try to employ.

Rewriting with these rules gives us the conclusion

$$\boxed{x \approx \widehat{c} \wedge x \neq \widehat{c} \to {}^*f(x) \approx \widehat{l_f} \wedge x \approx \widehat{c} \wedge x \neq \widehat{c} \to {}^*g(x) \approx \widehat{l_g}}^{\uparrow}.$$

Now we can instantiate the hypotheses using fertilisation and the theorem is proved.

LIM×

Let us now consider a proof for LIM× in non-standard analysis. The conjecture looks similar to that of LIM+, so one would expect the reasoning to be similar. The conjecture is written as

$$c, l_f, l_g : \mathbb{R} \quad f, g : \mathbb{R} \to \mathbb{R}$$
$$(\forall x \in {}^*\mathbb{R}.\ x \approx \widehat{c} \wedge x \neq \widehat{c} \to {}^*f(x) \approx \widehat{l_f}) \wedge (\forall x \in {}^*\mathbb{R}.\ x \approx \widehat{c} \wedge x \neq \widehat{c} \to {}^*g(x) \approx \widehat{l_g}) \vdash$$
$$\forall x \in {}^*\mathbb{R}.\ x \approx \widehat{c} \wedge x \neq \widehat{c} \to {}^*(\lambda z.\, f(z) \times g(z))(x) \approx \widehat{l_f \times l_g}.$$

After applying the rules $r\forall$, (4.47) and (4.35), we can embed the hypotheses in the conclusion to yield

$$x \approx \widehat{c} \wedge x \neq \widehat{c} \to \boxed{{}^*f(x) \times {}^*g(x)}^{\uparrow} \approx \boxed{\widehat{l_f} \times \widehat{l_g}}^{\uparrow}.$$

Now in order to reach the point at which fertilisation can complete the proof we need to use the wave rules

$$finite(Y) \wedge finite(B) \to \boxed{X \times A}^{\uparrow} \approx \boxed{Y \times B}^{\uparrow} \Rightarrow \boxed{X \approx Y \wedge A \approx B}^{\uparrow} \tag{5.3}$$
$$A \to \boxed{B \wedge C}^{\uparrow} \Rightarrow \boxed{A \to B \wedge A \to C}^{\uparrow}. \tag{5.4}$$

Using these rules is another example of **rippling out**, which allows completion of the proof by fertilisation. The side conditions to rule (5.3) are proved easily because every real number is finite, characterised by rule (4.38).

Chain Rule

The chain rule is a more complicated example from calculus. For this we need to use non-standard notions of differentiability introduced by theorem 5 of section 2.4.4. Note that for the theorems we present in this chapter, we are capable of yielding proof-plans using the non-standard characterisations for both uniform differentiability and non-uniform differentiability given in section 2.4.4. In this case we use the non-uniform differentiability. For ease of presentation we show the statement of the chain rule after applying rules (4.47),(4.50) and (4.35), which distribute the embedding and extension functions across multiplication and function composition:

$$x, d_1, d_2 : \mathbb{R} \quad f, g : \mathbb{R} \to \mathbb{R}$$
$$\forall h_1 \in {}^*\mathbb{R}.\ h_1 \approx 0 \wedge h_1 \neq 0 \to \frac{{}^*f(\widehat{g(x)}+h_1) - \widehat{f(g(x))}}{h_1} \approx \widehat{d_1} \tag{5.5}$$
$$\forall h_2 \in {}^*\mathbb{R}.\ h_2 \approx 0 \wedge h_2 \neq 0 \to \frac{{}^*g(\widehat{x}+h_2) - \widehat{g(x)}}{h_2} \approx \widehat{d_2} \vdash \tag{5.6}$$
$$\forall h \in {}^*\mathbb{R}.\ h \approx 0 \wedge h \neq 0 \to \frac{{}^*f({}^*g(\widehat{x}+h)) - \widehat{f(g(x))}}{h} \approx \widehat{d_1} \times \widehat{d_2}. \tag{5.7}$$

The first operation which we perform after $r\forall$ is to add the following formula to the hypotheses

$$^*g(\widehat{x}+h) - \widehat{g(x)} = 0 \vee {}^*g(\widehat{x}+h) - \widehat{g(x)} \neq 0. \tag{5.8}$$

We also set up the subgoal

$$\begin{gathered} x, d_1, d_2 : \mathbb{R} \quad f, g : \mathbb{R} \to \mathbb{R} \\ h \neq 0 \\ h \approx 0 \\ \forall h_1 \in {}^*\mathbb{R}.\ h_1 \approx 0 \wedge h \neq 0 \to \frac{{}^*f(\widehat{g(x)}+h_1) - \widehat{f(g(x))}}{h_1} \approx \widehat{d_1} \\ \forall h_2 \in {}^*\mathbb{R}.\ h_2 \approx 0 \wedge h_2 \neq 0 \to \frac{{}^*g(\widehat{x}+h_2) - \widehat{g(x)}}{h_2} \approx \widehat{d_2} \vdash \\ {}^*g(\widehat{x}+h) - \widehat{g(x)} \approx 0. \end{gathered} \tag{5.9}$$

In order to prove this subgoal, we first instantiate the universally quantified variable h_2 to h in the hypotheses. We now know that $h \approx 0$, and also that $\frac{{}^*g(\widehat{x}+h) - \widehat{g(x)}}{h} \approx \widehat{d_2}$. We introduce and use the conditional rewrite rule

$$finite(X) \wedge Y \approx 0 \wedge Y \neq 0 \ \to \qquad Z \approx 0 \ \Rightarrow \ \frac{Z}{Y} \approx X. \tag{5.10}$$

Recall that here $finite(X) \wedge Y \approx 0 \wedge Z \approx 0$ is the condition by which $Z \approx 0$ rewrites to $\frac{Z}{Y}$. The rewrite rule applies to the conclusion of (5.9) as its conditions are all satisfied since d_2 is real and hence by rule (4.38), finite.

It will be apparent why these steps (i.e. the case-split and the addition of a subgoal) were introduced when we come to complete this presentation of the proof. Once (5.8) has been added to the hypotheses of the chain rule, rule $l\vee$ is applied, and the proof splits into two branches which need to proved.

We first deal with the branch where $^*g(\widehat{x}+h) - \widehat{g(x)} \neq 0$. We can see that the hypotheses (5.5) and (5.6) do not embed separately in the conclusion (5.7) as we would like. By inspection we can see that we would like the term in one wave hole to be

$$\frac{{}^*g(\widehat{x}+h) - \widehat{g(x)}}{h}.$$

To achieve this, we need to use the lemma

$$\forall X \in {}^*\mathbb{R}.\ X \neq 0 \to \frac{Y}{Z} \approx \frac{Y}{X} \times \frac{X}{Z}. \tag{5.11}$$

This can be strengthened to an equality, but in order to complete the proof, we want to appeal to the transitivity of $\approx$. We use transitivity to rewrite the conclusion (5.7) and embed the

hypotheses. By inspection we can see that the X in the rewrite rule should be instantiated to ${}^*g(\widehat{x}+h)-\widehat{g(x)}$. The conclusion becomes

$$\forall h \in {}^*\mathbb{R}.\ h \approx 0 \wedge h \neq 0 \rightarrow \frac{{}^*f({}^*g(\widehat{x}+h)) - \widehat{f(g(x))}}{{}^*g(\widehat{x}+h)-\widehat{g(x)}} \times \frac{{}^*g(\widehat{x}+h)-\widehat{g(x)}}{h} \approx \widehat{d_1} \times \widehat{d_2}.$$

Now notice that we can rewrite terms in the conclusion using the equality

$$a = b + (a - b) \tag{5.12}$$

which is the rule is central to the reasoning pattern of **infinitesimal introduction**. We embed the hypotheses as desired yielding

$$h \approx 0 \wedge h \neq 0 \rightarrow \boxed{\frac{{}^*f(\widehat{g(x)}+{}^*g(\widehat{x}+h)-\widehat{g(x)})-\widehat{f(g(x))}}{{}^*g(\widehat{x}+h)-\widehat{g(x)}} \times \frac{{}^*g(\widehat{x}+h)-\widehat{g(x)}}{h}}^{\uparrow} \approx \boxed{\widehat{d_1} \times \widehat{d_1}}^{\uparrow}.$$

Now we can rewrite using (5.3) and (5.2), and yield the fully rippled conclusion

$$\boxed{h \approx 0 \wedge h \neq 0 \rightarrow \frac{{}^*f(\widehat{g(x)}+{}^*g(\widehat{x}+h)-\widehat{g(x)})-\widehat{f(g(x))}}{{}^*g(\widehat{x}+h)-\widehat{g(x)}} \approx \widehat{d_1} \wedge h \approx 0 \wedge h \neq 0 \rightarrow \frac{{}^*g(\widehat{x}+h)-\widehat{g(x)}}{h} \approx \widehat{d_2}}^{\uparrow}. \tag{5.13}$$

Once again in the last part of the proof we observe the reasoning pattern of **rippling out**. We can see that while the term in the blue wave hole matches a hypothesis exactly, the term in the red wave hole does not, as it has mismatching sinks. In order to complete the proof in this branch, we would like the wave hole to correspond to an instantiation of the remaining hypothesis. It is in this situation that piecewise fertilisation, as described in section 3.2.6, can take place. In this branch of the proof, we introduced the hypothesis

$${}^*g(\widehat{x}+h)-\widehat{g(x)} \neq 0$$

and before that we also introduced and proved the subgoal

$${}^*g(\widehat{x}+h)-\widehat{g(x)} \approx 0$$

Now piecewise fertilisation can succeed since the sinks in the red wave hole can be made to match.

Now let us consider the other branch in the proof, namely where ${}^*g(\widehat{x}+h)-\widehat{g(x)} = 0$. Then we can write $\widehat{g(x)}$ for ${}^*g(\widehat{x}+h)$ everywhere in the original goal using substitution rule *eq* from our sequent calculus. After $r\forall$ and $r\rightarrow$, the conclusion of the original goal (5.7) becomes

$$\frac{{}^*f({}^*g(\widehat{x}+h)) - \widehat{f(g(x))}}{h} \approx \widehat{d_1} \times \widehat{d_2}.$$

We know from the hypotheses that

$$h \neq 0\,, h \approx 0\,, \frac{{}^*g(\widehat{x}+h) - \widehat{g(x)}}{h} \approx \widehat{d_2}.$$

Substituting $\widehat{g(x)}$ for ${}^*g(\widehat{x}+h)$ we see that $\widehat{d_2} \approx 0$, and as $d_2 \in \mathbb{R}$, $d_2 = 0$. Now we can rewrite the conclusion using the substitution rule $\widehat{g(x)} = {}^*g(\widehat{x}+h)$, thereby reducing it to

$$\frac{0}{h} \approx \widehat{d_1} \times \widehat{d_2}.$$

We can use rule (4.53), together with rule (4.22) to obtain

$$0 \approx 0$$

which completes this branch of the proof. In this branch of the proof we see that we have had to use field rules in order to yield a proof-plan. Using the field rules constitutes a reasoning pattern, which we refer to as **arithmetical rearrangement**.

Rule (5.1): Uniform continuity of $+$

The continuity of $+$ is in itself a important theorem to prove. We cannot claim that the proof of LIM$+$ is complete until we have proved all of the lemmas involved, including the continuity of $+$ which is important to the proof. In standard analysis the proof of LIM$+$ is simple given the continuity of $+$ as a lemma. In our case we are faced with trying to prove

$$a, b, c, d : {}^*\mathbb{R}.$$
$$a \approx b$$
$$c \approx d \vdash$$
$$a + c \approx b + d.$$

We could annotate the conclusion to become

$$\boxed{a+c}^{\uparrow} \approx \boxed{b+d}^{\uparrow},$$

but we do not include annotation as it serves no purpose in the proof. We now use rule (5.12) to introduce new infinitesimal variables x and y, where $x = a - b$ and $y = c - d$. We then rewrite the conclusion to

$$(b + x) + (d + y) \approx b + d.$$

For the last part of the proof we rearrange the conclusion using our knowledge about the associative and commutative properties of $+$ to yield

$$b+d+x+y \approx b+d.$$

We know that $x \approx 0$, and $y \approx 0$, so we can use the rule (4.54) twice to yield the trivially provable conclusion

$$b+d \approx b+d.$$

Using rule (5.12) in the way introduced here is a reasoning pattern we refer to as **infinitesimal introduction**.

Rule (5.3): Continuity of $\times$

The continuity of $\times$ is crucial to the proof of LIM$\times$, and in non-standard analysis, we need to prove

$$a,b,c,d : {}^*\mathbb{R}$$
$$finite(b) \wedge finite(d)$$
$$a \approx b$$
$$c \approx d \vdash$$
$$a \times c \approx b \times d.$$

In order to prove this we use rule (5.12) again to introduce two new infinitesimal variables, $x = (a-b)$ and $y = (c-d)$ to rewrite the conclusion and yield

$$(b+x) \times (d+y) \approx b \times d.$$

We can rearrange this using the distribution laws for $\times$ to

$$(b \times d) + (x \times d) + (b \times y) + (x \times y) \approx b \times d$$

and now we can use rule (4.53), together with rules (4.17),(4.22) and (4.54) to reduce the conclusion to

$$b \times d \approx b \times d$$

which is completed by the reflexivity rule (4.28). The way that this rearrangemenet is achieved from the axioms is to write the axioms as rewrite rules. For example, in the case of axiom (4.53) we use the following conditional rewrite-rule:

$$finite(Y) \quad \rightarrow \qquad X \times Y \approx Z \times Y \Rightarrow X \approx Z$$

in the case of rule (4.54) we write the rewrite-rule

$$X + Y \approx Z \Rightarrow X \approx Z \wedge Y \approx 0.$$

This allows us to rewrite the goal and discharge the extra goals such as $x \approx 0$ and $y \approx 0$. Once again, the pattern of reasoning observed here is in **infinitesimal introduction**. Also in this proof-plan we observe **arithmetical rearrangement**.

Rule (5.2): Propositional rules

The rule (5.2) can be proved easily by the sequent rules presented in chapter 4. We conjecture the goal

$$A \to B, A \to C \vdash A \to B \wedge C$$

which is trivially proved using the sequent rules from the logic presented in section 4.

Rule (5.10): Auxiliary lemma for the chain rule

We need to prove the rule

$$finite(X), Y \approx 0, Y \neq 0, \frac{Z}{Y} \approx X \vdash Z \approx 0.$$

We can use rule (4.53) together with rule (4.22) to reach the goal

$$finite(X), Y \approx 0, Y \neq 0, X \times Y \approx 0, \frac{Z}{Y} \approx X \vdash Z \approx 0.$$

We know that Y is finite from rule (4.38). Now we can use rule (4.53) again to yield

$$finite(X), Y \approx 0, Y \neq 0, X \times Y \approx 0, \frac{Z}{Y} \times Y \approx X \times Y \vdash Z \approx 0.$$

We can then apply field rules (4.15), (4.16) and (4.24) to yield the conclusion

$$finite(X), Y \approx 0, Y \neq 0, X \times Y \approx 0, Z \approx X \times Y \vdash Z \approx 0$$

which is proved using the transitivity of $\approx$, given by rule (4.30).

Rule (5.11): Rule used in the chain rule proof

The associated theorem with this rule can be proved using simply the field rules given by rules (4.13), (4.14), (4.15), (4.16), (4.20), (4.21), (4.22), (4.23) and (4.24). We take the theorem

$$x, y, z : {}^*\mathbb{R}$$
$$z \neq 0 \vdash$$
$$\frac{x}{z} \times \frac{z}{y} \approx \frac{x}{y}.$$

Internally in λ*Clam* the left hand side of the conclusion is represented

$$(x \times z^{-1}) \times (z \times y^{-1}).$$

Using rule (4.15) twice, and then rules (4.16) and (4.24) we yield an identity.

5.2.2 Common reasoning patterns

The aim of developing plans for the proofs of the development set examples is to encapsulate the reasoning patterns that occur when performing these proofs. We mention in this section some of the patterns which are apparent in the proof, and how we represent them. We also discuss in section 5.3 how we incorporate these patterns in a set of plan-specifications.

For the more complicated method and patch specifications, we describe in words what happens and how we output new goals. There are some examples of the code written for these critics in section A.3.

Rippling out

One of the heuristics which underlies our approach to the automatic constructions of plans for the types of proof we present here is *rippling out*. In the proofs of the theorems presented, the first strategy is to rewrite the conclusion in such a way that it matches the hypotheses. In some of the proofs of section 5.2.1, this strategy works, and the proof can be completed by invoking the hypotheses as intended. In other proofs, we can see that some processing needs to be done in order to allow rippling to take place. In the first instance we want to be able to yield proof-plans for goals by embedding the hypotheses, rippling out fully, and invoking the hypotheses through strong fertilisation.

Infinitesimal introduction

As can be seen from the presentations of the proof-plans yielded for the development examples, we make use of rule (5.12) in order to yield proof-plans for some of the examples.

L'Hôpital suggested that it is possible to substitute infinitely close quantities. While this is sometimes possible, it is not always the case, since otherwise all infinitesimal values would reduce to 0. The purpose of rule (5.12) is to mimic equality substitution, and to reduce conclusions to the form

$$X \approx 0,\ Y \approx 0 \vdash B + X \approx B + Y \tag{5.14}$$

which can be proved by using rule (4.54) from the axiomatisation. When rippling cannot apply to a goal we try to yield proof-plans by reducing the conclusion to this form.

Arithmetical rearrangement

In order to be able to yield proof-plans for the examples in section 5.2.1, we need to apply field equations. We notice various patterns in which conclusions of goals can be rearranged. These are described in our presentation of the plan-specifications developed to account for the development set of theorems presented in section 5.2.1.

5.3 Plan-specifications

We initially constructed individual plan-specifications for each of the theorems in the development set. We show here how we amalgamate these plans into one initial plan-specification, and use critics to incorporate other general plan-specifications.

The plan-specifications are presented next as directed graphs. The names of the atomic methods are given in the boxes. If a method is capable of terminating with success, then the box is augmented with a horizontal line. Arrows between box indicate an application of a **then_meth** methodical. A directed loop indicates the application of a **repeat_meth** methodical.

5.3.1 Outermost plan-specification

The outermost plan-specification is written in $\lambda Clam$ as follows:

```
compound nsaconjectures nsa_top_meth_ripple_critics
(repeat_meth
  (then_meth tautology
  (then_meth (patch_meth set_up_ripple embed_critic_strat)
  (then_meth
  (repeat_meth
    (or_else_meth
      (patch_meth (wave_method outward R1) wave_critic_strat)
      (patch_meth (cond_wave_method outward R2) wave_critic_strat)))
  (then_meth (patch_meth fertilise fert_critic_strat))))))
  _
```

```
true.}
```

This is represented schematically by figure 5.1. As can be seen it works on the assumption that the conclusion should be rewritten until the hypotheses match. The idea is that the critics should react to failure of this strategy to suggest lemmas which can rewrite the conclusion. The following is a description of the steps shown in figure 5.1:

1. Tautology

Firstly the plan tries the `tautology` method, which applies all of the rules in the logic until a proof-plan is found, or leaves the goal completely unchanged. Since the `tautology` method only applies inference rules, it only solves propositional tautologies.

2. Embed Hypotheses

The hypotheses are embedded in the conclusion. If this fails then the embedding critic is employed to suggest a rule by which the conclusion can be rewritten so embedding can take place.

3. Ripple out

Once embedding has successfully taken place, the wave methods try to ripple the goal outwards. As discussed, we always try to ripple out, and never try to ripple in. Once rippling has successfully taken place, we re-calculate the embeddings of the hypotheses. If rippling fails then the wave critic is employed to suggest a lemma to unblock the rippling process.

4. Fertilise

Once the goal is completely rippled out, fertilisation is attempted. If this is not successful, the fertilisation critic is employed to suggest new subgoals which will allow piecewise fertilisation to take place.

5.3.2 Embedding patch plan-specification

When embedding fails, we employ a critic to analyse the failure and suggest a rule which can be used to rewrite the conclusion so that embedding will take place. We show the plan-specification we use for this in figure 5.2. The following is a description of the steps shown in the figure:

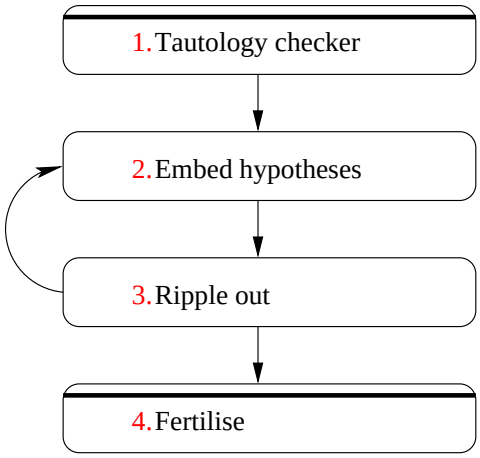

Figure 5.1: The outermost plan-specification

1. Apply embedding patch

The failure of embedding is analysed and a patch is suggested which outputs a lemma to use to rewrite the goal, and the rewritten conclusion. See Method 1 of section 5.4.1 for more details.

2. Lemma $\rightarrow$ Evaluation plan-specification

We attempt to find a proof-plan for the lemma using the evaluation plan-specification which we describe in section 5.3.5.

3. Goal $\rightarrow$ Outermost plan-specification

The goal is rewritten using the lemma, and then tackled using the outermost plan-specification.

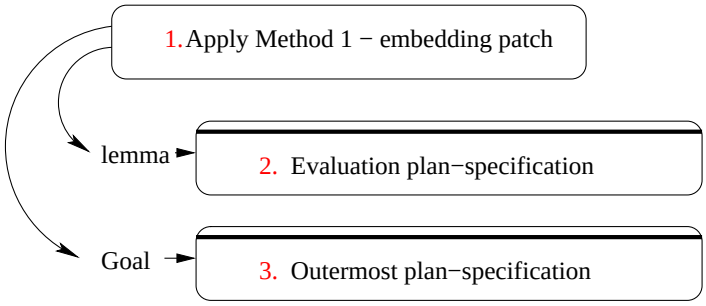

Figure 5.2: Embedding patch plan-specification

5.3.3 Wave patch plan-specification

When rippling out fails, we want to be able to speculate a lemma which will unblock the rippling process. We show the plan-specification we employ for this process in figure 5.3. We describe the steps that are shown in this figure as follows:

1. Apply wave patch

When rippling fails, the wave critic analyses the failure and speculates a lemma by which to rewrite the goal, and rewrites the goal using this lemma. See Method 2 of section 5.4.2 for more details.

2. Lemma $\rightarrow$ Evaluation plan-specification

The evaluation plan-specification described in section 5.3.5 is applied.

3. Goal $\rightarrow$ Outermost plan-specification

The rewritten goal is sent back to the outermost plan-specification.

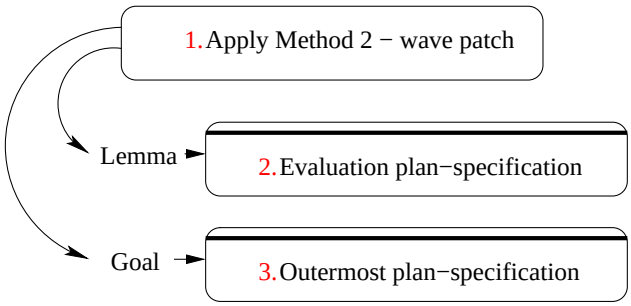

Figure 5.3: Wave patch plan-specification

5.3.4 Fertilisation patch plan-specification

When the goal is fully rippled out, but fertilisation cannot apply, we want to be able to add new subgoals which will allow piecewise fertilisation to apply (see section 3.2.6 for a description of piecewise fertilisation). This strategy is taken because we hypothesise that the reason that a fully rippled conclusion will not fertilise is due to mismatching sinks. The fertilisation patch plan-specification is shown in figure 5.4, and its steps can be described as follows:

1. Apply fertilisation patch

The failure of fertilisation prompts the critic patch to suggest facts which when added to the hypotheses would allow piecewise fertilisation to succeed.

2. Facts $\mathtt{C}$

The facts output by the fertilisation critic patch are both set as subgoals in step 4, and added to the hypotheses of the input goal in step 3.

3. $\mathtt{H} \cup \mathtt{C} \vdash G \rightarrow$ **Outermost plan-specification**

Here the facts in step 2 are used to augment the hypotheses of the initial goal. The outermost plan-specification is the applied to the new sequent.

4. For each $C_i \in \mathtt{C}$. $\mathtt{H}' \vdash C_i \rightarrow$ **Appropriate plan-specification**

At this point the facts in step 2 are interpreted as subgoals and are set as goals using the hypotheses from the fully rippled goal. The goal is then tackled with an appropriate plan-specification. See below for an explanation of this.

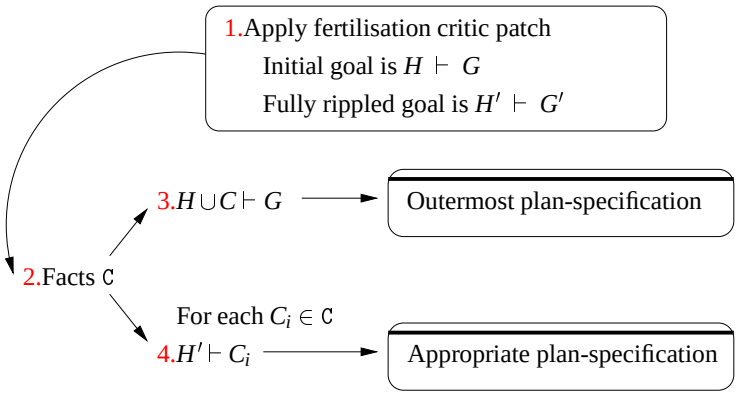

Figure 5.4: Fertilisation patch plan-specification

The original goal $\mathtt{H} \vdash G$ is the initial statement of the theorem with all of the universal quantifiers removed from G using the $r\forall$ inference rule from our logic. This is done to preserve any instantiations found for universally quantified variables in the hypotheses during the proof.

When $\lambda Clam$ sets up the subgoals $\mathtt{H}' \vdash C_i$ in step 4 of figure 5.4, it chooses which plan-specification to apply according to what form the subgoal has. If the subgoal has the form $L \neq R$ then $\lambda Clam$ adds $L = R$ to the hypotheses of the original goal, and applies the evaluation plan-specification. Thus, if the original goal is $\mathtt{H} \vdash G$, $\lambda Clam$ conjectures the new goal

$$\mathtt{H} \cup \{L = R\} \vdash G$$

and tackles it using the evaluation plan-specifications. In the other branch of the proof, the hypotheses of the original goal are augmented with $L \neq R$ yielding the goal

$$\mathtt{H} \cup \{L \neq R\} \vdash G.$$

This operation uses *em* rule from our sequent calculus, which introduces a case-split to the hypotheses. Then $l\vee$ is applied and the two branches described above are produced.

When the fertilisation critic patch outputs facts which are not of the form $L \neq R$, a subgoal is set up using the hypotheses from the fully rippled goal. Thus if a fact of the form $L \approx R$ is output by the fertilisation critic patch, for example, then, as indicated in step 4 of figure 5.4, the following subgoal is yielded:

$$\mathtt{H}' \vdash L \approx R.$$

This is tackled using the outermost plan-specification. $\lambda Clam$ uses the hypotheses from the fully rippled goal, $\mathtt{H}' \vdash G'$, because they may contain information yielded during the attempted proof of the original goal, $\mathtt{H} \vdash G$. Using facts both to augment hypotheses and to be set as subgoals is an example of the use of the *cut* rule of inference from our sequent calculus.

The fact $L \approx R$ is added to the hypotheses of the original goal, $\mathtt{H} \vdash G$, along with any other facts which were output by the fertilisation critic patch. For the two examples of subgoal mentioned above: $L \approx R$ and $L \neq R$, $\lambda Clam$ yields the new goal

$$\mathtt{H} \cup \{L \approx R\} \cup \{L \neq R\} \vdash G$$

which is tackled using the outermost plan-specification.

5.3.5 Evaluation plan-specification

The evaluation plan-specification shown in figure 5.5 shows how we tackle goals which cannot be completed using the outermost plan-specification. This is very important as it incorporates the patterns of reasoning we see in proofs such as the continuity of $\times$ given in section 5.2.1. The following is a description of the steps shown in figure 5.5

1. Simulating substitution

This is described in section 5.4 by Method 3. We simulate substitution for the infinitely close relation by using rule (5.12) to replace terms.

2. Multiplying through by denominators

We perform some simple field operations on the conclusion of a goal to put it into a chosen normal form, as described by Method 4 in section 5.4.

3. Using equality substitution

When appropriate equalities are found in the hypotheses we apply substitution to the goal, as described by Method 5.

4. Simplification

We apply simplification rules to the goal as described in Method 6.

5. Eliminate infinitesimal terms

As described in Method 7, we eliminate infinitesimal quantities from either side of the infinitely close relation.

6. Goal trivially provable

We use some rules to determine whether the goal is trivially provable, as described in Method 8.

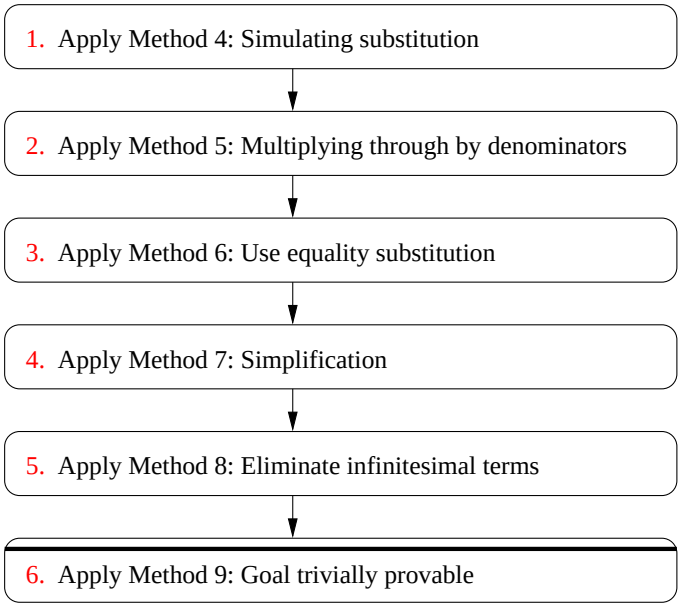

Figure 5.5: Evaluation plan-specification

5.4 Methods and Critics

We describe the methods and critics we have written to allow the plan-specifications of the previous section to produce proof-plans. We start with a presentation of the critics developed,

and their associated patches, and then go on to describe the methods which are used in the evaluation plan-specification shown in figure 5.5. Our presentation of methods includes the input goal sequent, all of the preconditions and postconditions, and the output goal sequent. We denote sets by true type symbols, such as H, and terms in standard font, such as G, with functions in calligraphic font such as $\mathcal{F}$. Where substitutions are to be applied, we use the symbol $\circ$ and apply sets of substitutions to terms.The output goal sequent can be a single goal sequent, or can be constructed from pairs of goal sequents.

5.4.1 Embedding critic

We hypothesise that all proofs introduced in this chapter should end by invoking the hypotheses using fertilisation. Although this is not always possible, we want the ability to embed the hypotheses and then use the coloured rippling machinery to ripple out as far as possible. Let us consider the example of speculating rule (5.11) used in proving the chain rule. In particular, let us consider again the blocked conclusion encountered in the proof of the chain rule

$$h \approx 0 \wedge h \neq 0 \rightarrow \frac{{}^{*}f({}^{*}g(\widehat{x+h})) - \widehat{f(g(x))}}{h} \approx \widehat{d_1} \times \widehat{d_2}.$$

We cannot embed the hypotheses (5.5) and (5.6) in the conclusion, and so the method cannot employ coloured rippling. The critic here is attached to the set_up_ripple method, which is called every time before rippling. The patch suggested is described schematically by Method 1.

We impose strict conditions on the ability to embed hypotheses so that this critic can suggest rules that allow rippling to occur. For example, we state that for any term $A \approx B$ in the hypotheses, both A and B must embed fully in the conclusion for rippling to take place. Clearly this is a very stringent condition but it allows the outermost plan-specification to deal with the most general structure of the proof, and the embedding critic to find the rules which manipulate the term structure of the conclusion to allow this condition to be met. Any resulting rule will then be tackled with the evaluation plan described in section 5.3.5.

The general idea underlying the patch for the embedding critic shown by Method 1 is to speculate a rule by hypothesising that the infinitely close relation works in some way like equality. Thus lemmas are speculated by replacing terms in the conclusion with infinitely close terms in the hypotheses. While this is not necessarily correct, it is a heuristic which guides us towards finding the shape of a lemma which can be used to rewrite the goal, and allow embedding to take place. In order to construct the lemma, we must reason about subterms of

universally quantified variables. This means that we must guess instantiations for the variables which are universally quantified in the hypotheses. We employ a simple equational unification algorithm which incorporates the equality

$$A = B + (A - B)$$

which is crucial to the reasoning pattern of **infinitesimal introduction**. The way that the lemma is stated, and then subsequently used as a rewrite rule follows the same technique as that for the wave critic, which we present later in section 5.4.2.

Method 1 Embedding critic patch

Input: Goal: $\mathtt{H} \vdash A \to B \approx C$

Conditions:

Find terms $L_1 \approx R_1, ..., L_n \approx R_n$ in hypotheses

Unify C with $\mathcal{F}(R_1, ..., R_n)$ instantiating $\mathcal{F}$

State Lemma as $B \approx \mathcal{F}(L_1, ..., L_n)$

Output: Lemma and Rewritten Goal

5.4.2 Wave Critic

In this work we are interested in rippling out fully, so that the hypotheses can apply. In some cases sinks can accumulate terms using rippling in, but this normally happens in recursive proofs. As our current theories do not involve induction, we do not want to ripple in, and we attempt to ripple out only in the proof-plan. When the wave method fails, it indicates that a rewrite rule cannot be found to continue rippling out, and a critic is fired to speculate the missing lemma. Coloured rippling allows us to analyse what the form of the rewrite should be. The preconditions to the built-in wave method are

```
given goal H |- G with embedding E:
rewrite goal H |- G with rule R to give goal H |- G'
calculate embedding E' of H in G'
check that measure reduces from E to E'.
```

The critic fires if the measure does not reduce. This means that the only wave rule applicable is one which does not ripple out but in, since rippling out was tried in every possible way before rippling in attempted. Now the critic's job is to find a suitable rule that can rewrite the

conclusion. We employ the procedure outlined by Method 2. For the code which describes this patch see section A.3.

Method 2 Lemma speculation method for wave-critic patch

Input: Goal: $\mathtt{H} \vdash G$

Conditions:

Case: if only one wave hole per hypothesis:

Join each wave hole to nearest shared term

Speculate rule which allows new wave holes to match more of hypotheses

Case: if more than one wave hole per hypothesis:

Join wave holes together using functors in skeleton

Speculate rule which allows new wave holes to match more of hypotheses

Output: Lemma and Rewritten Goal

As an example of this method at work, consider the proof of $LIM\times$ given in section 5.2.1. We show how the patch specified by Method 2 calculates a lemma which can be used to allow rippling to proceed. In particular we show how meta-variables are used to represent conditions to lemma which can only be discovered during the execution of the proof-plan. We start with the blocked conclusion

$$x \approx \widehat{c} \wedge x \neq \widehat{c} \rightarrow \boxed{{}^*f(x) \times {}^*g(x)}^{\uparrow} \approx \boxed{\widehat{l_f} \times \widehat{l_g}}^{\uparrow}.$$

Firstly the method preconditions find the terms in the wave holes using the embedding for each hypothesis. In this presentation we represent the binding of each term to a hypothesis using colour:

$${}^*f(x) \quad {}^*g(x) \quad \widehat{l_f} \quad \widehat{l_g}.$$

Now the method attempts to join the like coloured terms together using the information from the hypotheses and the conclusion. In the first (red) hypothesis, the method records the formula ${}^*f(x) \approx \widehat{l_f}$, and from the second (blue) hypothesis, it records the formula ${}^*g(x) \approx \widehat{l_g}$. Now $\lambda Clam$ generalises each term to a new variable, determining its type, and conjectures the goal

$$a, b, c, d : {}^*\mathbb{R}$$
$$\mathcal{M}(a, b, c, d)$$
$$\vdash a \approx b \wedge c \approx d \rightarrow a \times c \approx b \times d.$$

Here $\mathcal{M}$ is a higher-order variable which is initially uninstantiated. During the construction of the proof-plan for the lemma, it is instantiated to a function of the variables a, b, c and

d. We discuss this issue further in section 5.4.5, and a proof of this theorem is presented in section 5.2.1. We can see from this proof that $\mathcal{M}$ becomes instantiated to $\lambda a,b,c,d.\ finite(b) \wedge finite(d)$. Once this goal has been proved, we can add wave rule (5.3) to the system, and it can be used in yielding proof-plans for other theorems. We choose to represent the lemma as a conditional wave rule with $\mathcal{M}(a,b,c,d)$ as the condition.

5.4.3 Fertilisation critic

We hypothesise that the failure of strong fertilisation is due to the existence of mismatching sinks in the wave holes. In this case we must set new subgoals which allow piecewise fertilisation to succeed. In some cases these subgoals are not complicated to prove, but in others we need a specific strategy for proving difficult subgoals. We employ a case-split in the instance where must prove a subgoal of the form $A \neq B$, adding $A = B \vee A \neq B$ to the hypotheses of the goal, and splitting it into two branches.

As can be seen from the proof of the chain rule in section 5.2.1, a case-split must be performed at the beginning of the proof. This operation is characterised by rule *em* from the logic given in table 4.1. This rule is non-terminating, and so its application must be carefully controlled. We choose to control it by limiting its application just to critics, which react to the failure of the strong fertilisation method. If the wavefronts are all fully rippled out, and the strong fertilisation method does not apply, then the sinks do not match. In this instance we attempt such a case split.

The preconditions to the built-in strong fertilisation method are that all of the conjuncts of the goal should match a hypothesis exactly. The preconditions to the method are

```
given goal H |- G with embedding
Conclusion G is fully rippled
Sinks in G match in H.
```

The critic reacts to the failure of the sinks to match. The method for the fertilisation critic patch is used by the plan-specification in 5.4.

As an example of the fertilisation patch at work, let us consider again the proof of the chain rule. The orignal conclusion of the chain rule (with the hypotheses abbreviated to `H`), and $r\forall$ applied, is

$$h : {}^*\mathbb{R}.\ \texttt{H} \vdash\ \ h \approx 0 \wedge h \neq 0 \rightarrow \frac{{}^*f({}^*g(\widehat{x+h})) - \widehat{f(g(x))}}{h} \approx \widehat{d_1} \times \widehat{d_2}. \tag{5.15}$$

Once the conclusion has been fully rippled $\lambda Clam$ yields

$$\mathtt{H}' \vdash h \approx 0 \wedge h \neq 0 \rightarrow \frac{{}^*f(\widehat{g(x)}) + {}^*g(\widehat{x}+h) - \widehat{g(x)}) - \widehat{f(g(x))}}{{}^*g(\widehat{x}+h) - \widehat{g(x)}} \approx \widehat{d_1} \wedge h \approx 0 \wedge h \neq 0 \rightarrow \frac{{}^*g(\widehat{x}+h) - \widehat{g(x)}}{h} \approx \widehat{d_2} \quad .$$

In the fully rippled conclusion, the fertilisation patch notices that the term in the red wave hole has mismatching sinks. The facts which are output by the critic patch are

$${}^*g(\widehat{x}+h) - \widehat{g(x)} \neq 0. \tag{5.16}$$

$${}^*g(\widehat{x}+h) - \widehat{g(x)} \approx 0 \tag{5.17}$$

Following the plan-specification in figure 5.4 we set up three new goals in order to allow piecewise fertilisation to succeed:

- Firstly, for the fact (5.16), $\lambda Clam$ specifies the subgoal

  $$h : {}^*\mathbb{R}\ , \mathtt{H},\ {}^*g(\widehat{x}+h) - \widehat{g(x)} = 0 \vdash {}^*\mathbb{R}.\ h \approx 0 \wedge h \neq 0 \rightarrow \frac{{}^*f({}^*g(\widehat{x}+h)) - \widehat{f(g(x))}}{h} \approx \widehat{d_1} \times \widehat{d_2}$$

  which is tackled using the evaluation plan-specification.

- Secondly, for the fact (5.17), $\lambda Clam$ specifies the subgoal

  $$\mathtt{H}' \vdash {}^*g(\widehat{x}+h) - \widehat{g(x)} \approx 0$$

  which is tackled using the evaluation plan-specification.

- $\lambda Clam$ specifies a restatement of the original goal:

  $$h : {}^*\mathbb{R},\ \mathtt{H},\ {}^*g(\widehat{x}+h) - \widehat{g(x)} \neq 0,\ {}^*g(\widehat{x}+h) - \widehat{g(x)} \approx 0 \vdash h \approx 0 \wedge h \neq 0 \rightarrow \frac{{}^*f({}^*g(\widehat{x}+h)) - \widehat{f(g(x))}}{h} \approx \widehat{d_1} \times \widehat{d_2}.$$

  This is tackled using the outermost plan-specification.

5.4.4 Methods for the evaluation plan-specification

We present here the methods employed in yielding proof-plans via the evaluation plan-specification in figure 5.5. If any conditional rewrite rules are used in the methods, the conditions are set as subgoals, and attempted by this plan also.

Infinitesimal introduction

When rippling is not possible, we need to make use of the hypotheses in a different way than by just instantiating them with the terms in the conclusion. Notice in the proofs presented in section 5.2.1 that when rippling was not used, the rule (5.12):

$$A = B + (A - B)$$

was very often central to the proof. This rule is used so often in the proofs because it mirrors the behaviour of substitution for equality, but with the $\approx$ operator. When we know from the hypotheses that $a \approx b$, we can simulate the substitution behaviour of equality by writing $b + (a - b)$ for a in the conclusion. This leaves us with terms $a - b$ which we know to be infinitesimal, and so can disregard under certain circumstances. The proof of the continuity of $+$ given in section 5.2.1 shows an example of how this technique can yield a proof.

Method 3 describes the approach used to carry out the rearrangement of the conclusion using rule 5.12. Here we use the notation $\circ$ when applying subsitutions to terms. For example $G \circ S$ denotes the goal G under the subsitution S.

Method 3 Use of infinitesimal introduction in simulating substitution

---

Input: Goal: $\mathtt{H} \vdash G$

Conditions:

$$\mathtt{X} = \{Y \approx 0 \wedge B - A = Y \,:\, A \approx B \in \mathtt{H}\}$$

$$\mathtt{S} = \{(B + Y)/A \,:\, Y \approx 0 \wedge B - A = Y \in \mathtt{X}\}$$

Output: Goal: $\mathtt{H} \cup \mathtt{X} \vdash G \circ \mathtt{S}$

Multiplying through by denominators

One more important technique which has been used in the proofs is that of simple field operations to collect terms. As can be seen from theorem (5.1) for example, the field equations are needed to rearrange the terms in conclusions so that we can get to a point where we have the form given by (5.14):

$$X \approx 0,\ Y \approx 0 \vdash B + X \approx B + Y.$$

We can then complete the proof using other rules from the axiomatisation.

The advantage non-standard analysis gives us, is being able to always prove conjectures of the form shown by (5.14). When working with the reals we are not able to exploit the concept of infinitely close, which encapsulates the notion of a limit in standard analysis.

The first stage in the process of yielding a form for the conclusion which looks like (5.14) is to expand all terms using the distributive rule (4.17) from the axiomatisation, and its commutative counterpart, and to multiply through by denominators. Method 4 describes this process. It is important to notice that we can only multiply through by finite quantities to preserve the $\approx$ relation.

Method 4 Multiplying through by denominators

Input: Goal: $\mathtt{H} \vdash L \approx R$

Conditions:

Exhaustively Apply rules:

$r \rightarrow, l \rightarrow$

$A \times (B+C) \rightarrow (A \times B) + (A \times C)$

$(A+B) \times C \rightarrow (A \times C) + (B \times C)$

Multiply R through by all finite denominators of L

Multiply L through by all finite denominators of R

Output: Goal: $\mathtt{H} \vdash L \approx R$

Use equality substitution

We describe the method which replaces values according to equalities which arise in hypotheses in method 5. This method applies substitution in one direction, and replaces terms in the conclusion, setting the conclusion up for simplification.

Method 5 Using equality substitution

Input: Goal: $\mathtt{H} \vdash G$

Conditions:

$\mathtt{S} = \{A/B \ : \ A = B, A - B = 0 \in \mathtt{H}\}$

Output: Goal: $\mathtt{H} \circ \mathtt{S} = G \circ \mathtt{S}$

Simplification

Method 6 uses the identity rules from the axiomatisation (4.21),(4.14),(4.16), (4.24) and (4.22). In this method, duals of each rule are used to include symmetry. For example, the rules $0 + x \Rightarrow x$ and $x + 0 \Rightarrow x$ are included.

Method 6 Simplifying the goal

Input: Goal: $\mathtt{H} \vdash G$

Conditions:

Exhaustively apply rules to $\mathtt{H}$ and G

$0 + X \Rightarrow X$

$(-X) + X \Rightarrow 0$

$X \neq 0 \rightarrow X \times X^{-1} \Rightarrow 1$

$1 \times X \Rightarrow X$

$0 \times X \Rightarrow 0$

$\widehat{X} \approx \widehat{Y} \Rightarrow X = Y$

To G until no more apply yielding G'

Output: Goal: $\mathtt{H} \vdash G'$

Eliminate infinitesimal terms

We introduce Method 7 which will recognise the structure of a goal which has the shape described by the form given by (5.14). This is important since we can employ rules (4.54) and (4.53) from our axiomatisation to eliminate these terms. It replaces all infinitesimal terms in a sum by zero if the sum is on one side of the infinitesimally close relation $\approx$.

Method 7 Finding infinitely close quantities in conclusions

Input: Goal: $\mathtt{H} \vdash G$

Conditions:

$G = T_1 + ... + T_i \approx T_{i+1} + ... + T_n$

$S = \{0/T_i \ : \ T_i = (A \times B), A \approx 0 \in \mathtt{H}, finite(B) \in \mathtt{H}\}$

Output: Goal: $\mathtt{H} \vdash G \circ \mathtt{S}$

Goal trivially provable

Method 8 shows the rules we can use in order to complete a proof. We use the tautology method, as described in section 5.3, and rules (4.38), (4.10) and (4.28) from the axiomatisation. Note that here we include rules to show finiteness and non-zero properties which are set as subgoals for some rules. We discuss this issue further in the next section.

Method 8 Rules for proving goals

Input: Goal: $\mathtt{H} \vdash G$

Conditions:

Apply Rules:

$X = X \Rightarrow \top$

$X \approx X \Rightarrow \top$

$finite(\widehat{X}) \Rightarrow \top$

Apply `tautology` method

Output: Branch closed

5.4.5 Subgoals in the evaluation plan-specification

Some conditional rewrite rules are used in the evaluation plan-specification, such as (4.16)

$$X \neq 0 \rightarrow X \times X^{-1} \Rightarrow 1.$$

The rewriting process can go ahead if the condition exists in the hypotheses; if not then the condition is set as a subgoal which is tackled using the evaluation plan-specification.

When lemmas are speculated by the wave critic or the embedding critic, a higher-order variable is placed in the hypotheses. During the construction of the proof-plan for the lemma, this higher-order variable is instantiated to a conjunction of all the unprovable conditions to the rewrite-rules that were used. This can be seen in the proof-plan described for the continuity of $\times$ in section 5.2.1, where finiteness conditions must be imposed on the variables.

5.5 Test set

We present the examples used for our test set, describing whether the plan-specifications presented in section 5.3 successfully yielded proof-plans, and if not, what work still remained to be done interactively.

5.5.1 Continuity of $-$

$\lambda Clam$ constructs a proof-plan for the following theorem

$$a, b, c, d : {}^*\mathbb{R}$$

$$a \approx b$$

$$c \approx d \vdash$$
$$a - c \approx b - d.$$

The conclusion is annotated to become

$$\boxed{a - c}^{\uparrow} \approx \boxed{b - d}^{\uparrow}.$$

At this point rippling cannot continue and so the lemma speculation machinery fires via the wave critic, and the resulting lemma is proved using the plan-specification shown in figure 5.3. Using rule (5.12) twice the conclusion is rewritten to

$$(b + (a - b)) - (d + (c - d)) \approx b - d.$$

Using the evaluation plan-specification this is rearranged and the proof-plan completed by the reflexivity of $\approx$ (4.28) incorporated into Method 8.

5.5.2 Continuity of $/$

We attempt to construct a proof-plan for the conjecture

$$a, b, c, d : {}^*\mathbb{R}.$$
$$\mathcal{M}(a, b, c, d)$$
$$d \neq 0$$
$$a \approx b$$
$$c \approx d \vdash$$
$$\frac{a}{c} \approx \frac{b}{d}$$

$\lambda Clam$ now annotates the conclusion to become

$$\boxed{\frac{a}{c}}^{\uparrow} \approx \boxed{\frac{b}{d}}^{\uparrow}.$$

At this point rippling cannot continue and so the lemma speculation machinery fires via the wave critic. The wave critic patch uses the evaluation plan-specification to yield a proof-plan for the goal. It introduces infinitesimal variables $x = (a - b)$ and $y = (c - d)$ and leaves the following lemma to be proved:

$$\frac{b + x}{d + y} = \frac{b}{d}.$$

This is rearranged by Method 4, which cross-multiplies with the denominators, and then solved using Methods 7 and 8 from the evaluation plan-specification. $\mathcal{M}$ becomes instantiated to $finite(d) \wedge finite(d + (c - d))$ during the execution of the proof-plan.

This is in fact an incorrect proof, since the correct conditions for the theorem to hold are $a \approx b \ \wedge \ c \approx d \ \wedge \ d \not\approx 0$. The problem here is that there is a fundamental error in Method 4, which incorrectly imposes finiteness properties when in fact the correct properties to impose are that divisors are not infinitely small. This is discussed further in section 5.7.

5.5.3 LIM −

We write this theorem as:

$$c\,, l_f\,, l_g : \mathbb{R} \quad f, g : \ \mathbb{R} \to \mathbb{R}$$
$$(\forall x \in {}^*\mathbb{R}. x \approx \widehat{c},\ \ x \neq \widehat{c} \to {}^* f(x) \approx \widehat{l_f}) \wedge (\forall x \in {}^*\mathbb{R}. x \approx \widehat{c} \wedge x \neq \widehat{c} \to {}^* g(x) \approx \widehat{l_g}) \vdash$$
$$\forall x \in {}^*\mathbb{R}. x \approx \widehat{c} \wedge x \neq \widehat{c} \to {}^*(\lambda z.\, f(z) - g(z))(x) \approx \widehat{l_f - l_g}$$

The outermost plan-specification and the lemma speculation machinery successfully construct a complete proof-plan for this theorem. The lemma speculation machinery speculates the rule shown in section 5.5.1, and uses it to complete the proof of this theorem.

5.5.4 LIM /

We write this theorem as:

$$l_g \neq 0 \ \wedge \ \forall x \in \mathbb{R}.\ g(x) \neq 0 \ \wedge \lim_{x \to c} f(x) = l_f \wedge \lim_{x \to c} g(x) = l_g \vdash \lim_{x \to c} \frac{f(x)}{g(x)} = \frac{l_f}{l_g}$$

Notice that we restrict the function g to be non-zero everywhere, and define the limit point l_g to be non-zero, since we want the conclusion to be well-defined. When the theorem is written out formally using the non-standard characterisation of a limit, the following conjecture is yielded:

$$c\,, l_f\,, l_g : \mathbb{R} \quad f, g : \ \mathbb{R} \to \mathbb{R}$$
$$l_g \neq 0$$
$$\forall x \in {}^*\mathbb{R}.\ \ {}^* g(x) \neq 0$$
$$(\forall x \in {}^*\mathbb{R}.\ x \approx \widehat{c} \wedge \ \ x \neq \widehat{c} \to {}^* f(x) \approx \widehat{l_f}) \wedge (\forall x \in {}^*\mathbb{R}.\ x \approx \widehat{c} \wedge x \neq \widehat{c} \to {}^* g(x) \approx \widehat{l_g}) \vdash$$
$$\forall x \in {}^*\mathbb{R}.\ x \approx \widehat{c} \wedge x \neq \widehat{c} \to {}^*(\lambda z. \tfrac{f(z)}{g(z)})(x) \approx \widehat{\tfrac{l_f}{l_g}}$$

We apply the outermost plan-specification shown in figure 5.1. The conjecture becomes blocked when the ripple out method is attempted and the wave critic fires. The patch is to speculate the rewrite rule

$$\mathcal{M}(a,b,c,d) \rightarrow \frac{a}{c} \approx \frac{b}{d} \Rightarrow a \approx b \wedge c \approx d$$

whose proof was shown in section 5.5.2. The instantiation for $\mathcal{M}$ which is calculated is $finite(d) \wedge finite(d+y)$, where $y = c - d$. The outermost plan-specification uses this lemma to rewrite the conclusion. From this point the goal is fully rippled out and fertilisation applies and a complete proof-plan is yielded.

$\lambda Clam$ is left to satisfy the conditions of the rewrite rule. In particular, it must show that $finite(\widehat{l_g})$ and $finite({}^*g(x))$ where x is the variable introduced by the $r\forall$ rule. These subgoals are attempted by the evaluation plan-specification. Method 8 produces a proof-plan for the $finite(\widehat{l_g})$ subgoal since l_g is real. However, it does not yield a proof-plan for the subgoal $finite({}^*g(x))$. We need to yield a proof-plan for this goal interactively with the system.

It is easy to see how the condition $finite({}^*g(x))$ can be proved. With hindsight, we should have introduced a method which dealt specifically with finiteness conditions. Given the instantiation we know for the universally quantified variables in the hypotheses, and the rules we introduced for the predicate subtype `finite` in the axiomatisation– namely (4.42), (4.43), (4.44) and (4.45), it would be easy to implement such a method.

We note at this point, as in the presentation of the proof-plan for the continuity of / in section 5.5.2 that the lemma

$$\mathcal{M}(a,b,c,d) \rightarrow \frac{a}{c} \approx \frac{b}{d} \Rightarrow a \approx b \wedge c \approx d.$$

only applies if $d \not\approx 0$. This can be deduced since $l_g \neq 0$, yet l_g is a real number, hence $l_g \not\approx 0$. It is in fact not necessary for this theorem to impose the condition $\forall x \in {}^*\mathbb{R}.\ {}^*g(x) \neq 0$. These are flaws in the implementation of Method 4, and is further discussed in section 5.7.

5.5.5 Product Rule

For ease of presentation we show the product rule conclusion after interactively applying rules (4.34), (4.35) and (4.47), which distributes the embedding and extension functions across addition and multiplication:

$$x, d_1, d_2 : \mathbb{R} \qquad f, g : \mathbb{R} \rightarrow \mathbb{R}$$
$$\forall h \in {}^*\mathbb{R}.\ h \approx 0 \wedge h \neq 0 \rightarrow \frac{{}^*f(\hat{x}+h) - \widehat{f(x)}}{h} \approx \widehat{d_1}$$

$$\forall h \in {}^*\mathbb{R}.\ h \approx 0 \wedge h \neq 0 \rightarrow \frac{{}^*g(\widehat{x}+h)-\widehat{g(x)}}{h} \approx \widehat{d_2} \vdash$$
$$\forall h \in {}^*\mathbb{R}.\ h \approx 0 \wedge h \neq 0 \rightarrow \frac{{}^*f(\widehat{x}+h)\times{}^*g(\widehat{x}+h)-\widehat{f(x)}\times\widehat{g(x)}}{h} \approx (\widehat{d_1}\times\widehat{g(x)})+(\widehat{f(x)}\times\widehat{d_2}). \quad (5.18)$$

We apply the outermost plan-specification to this goal. Initially, the embedding mechanism does not succeed as it cannot find an embedding according to the stringent conditions placed upon it. The planner therefore fires the embedding critic plan, which suggests rewriting the conclusion to

$$h \approx 0 \wedge h \neq 0 \rightarrow \boxed{\left(\frac{{}^*f(\widehat{x}+h)-\widehat{f(x)}}{h}\right)\times\widehat{g(x)}+\widehat{f(x)}\times\frac{{}^*g(\widehat{x}+h)-\widehat{g(x)}}{h}}^{\uparrow} \approx \boxed{d_1\times\widehat{g(x)}+\widehat{f(x)}\times d_2}^{\uparrow}.$$

In order to do this, the embedding critic patch shown in Method 1 calculates the formula

$$\frac{{}^*f(\widehat{x}+h)\times{}^*g(\widehat{x}+h)-\widehat{f(x)}\times\widehat{g(x)}}{h} \approx \left(\frac{{}^*f(\widehat{x}+h)-\widehat{f(x)}}{h}\right)\times\widehat{g(x)}+\widehat{f(x)}\times\left(\frac{{}^*g(\widehat{x}+h)-\widehat{g(x)}}{h}\right).$$

This is used to guide the rewriting of the conclusion (5.18), so that embedding can take place. The associated lemma which is speculated is

$$a,b,c,d,e : {}^*\mathbb{R}$$
$$\mathcal{M}(a,b,c,d,e) \vdash$$
$$\frac{(a\times b)-(c\times d)}{e} \approx \left(\frac{a-c}{e}\times d\right)+\left(c\times\frac{b-d}{e}\right). \quad (5.19)$$

The evaluation plan-specification does not yield a proof-plan for this lemma, as the proof requires factorisation. As this lemma will help to produce a proof-plan for the product rule, we interact with $\lambda Clam$, and add the following rewrite rule to the system:

$$A \approx C \wedge B \approx D \wedge E \approx 0 \rightarrow \quad \frac{(A\times B)-(C\times D)}{E} \Rightarrow \left(\frac{A-C}{E}\times D\right)+\left(C\times\frac{B-D}{E}\right)$$

thus instantiating $\mathcal{M}(a,b,c,d,e)$ in the lemma to $a \approx c \wedge b \approx d \wedge e \approx 0$. The application of this lemma can be seen intuitively to be correct in the context of the product rule by considering the equality

$$\frac{(A\times B)-(C\times D)-(A-C)\times(B-D)}{E} = \frac{((A-C)\times D)+(C\times(B-D))}{E}$$

if $E \neq 0$. If $A \approx B$ and $C \approx D$, then

$$\frac{(A\times B)-(C\times D)}{E} \approx \frac{((A-C)\times D)+(C\times(B-D))}{E}$$

since $\frac{(A-C)\times(B-D)}{E} \approx 0$. Now we can appeal to the transitivity of $\approx$, to rewrite the conclusion (5.18). The goal is rewritten to

$$x, d_1, d_2 : \mathbb{R} \quad f, g : \mathbb{R} \to \mathbb{R} \quad h : {}^*\mathbb{R}$$
$$\forall h \in {}^*\mathbb{R}.\ h \approx 0 \wedge h \neq 0 \to \frac{{}^*f(\widehat{x}+h) - \widehat{f(x)}}{h} \approx \widehat{d_1}$$
$$\forall h \in {}^*\mathbb{R}.\ h \approx 0 \wedge h \neq 0 \to \frac{{}^*g(\widehat{x}+h) - \widehat{g(x)}}{h} \approx \widehat{d_2} \vdash$$
$$h \approx 0 \wedge h \neq 0 \to \boxed{\left(\frac{{}^*f(\widehat{x}+h) - \widehat{f(x)}}{h}\right) \times \widehat{g(x)} + \widehat{f(x)} \times \frac{{}^*g(\widehat{x}+h) - \widehat{g(x)}}{h}}^{\uparrow} \approx \boxed{d_1 \times \widehat{g(x)} + \widehat{f(x)} \times d_2}^{\uparrow}.$$

The wave critic then speculates the rule

$$finite(X) \wedge finite(Y) \to \boxed{A \times X + Y \times B}^{\uparrow} \approx \boxed{C \times X + Y \times D}^{\uparrow} \Rightarrow \boxed{A \approx C \wedge B \approx D}^{\uparrow} \quad (5.20)$$

which leads to a completion of the construction of a plan for the product rule using the outermost plan-specification. We find a proof-plan for the lemma associated with this rule automatically using the evaluation plan-specification.

As mentioned, part of this proof-plan needs to be constructed interactively. The embedding critic patch successfully suggests a rule by which to rewrite the conclusion (5.18), but the evaluation plan-specification is not capable of constructing a proof-plan for the speculated lemma. We therefore add this lemma to the system as a rewrite rule, so that a proof-plan for the product rule can be completed.

A brief analysis shows that $\lambda Clam$ is unable to find a proof-plan for lemma (5.19) through the evaluation plan-specification because the proof of this lemma requires factorisation which was not accounted for in the design of our methods and critics.

5.5.6 Extra limit conjecture

One final conjecture which we tested on the system is

$$\forall x \in {}^*\mathbb{R}.\ x \neq \widehat{c} \wedge x \approx \widehat{c} \to {}^*f(x) \approx \widehat{l} \vdash \forall y \in {}^*\mathbb{R}.\ y \neq 0 \wedge y \approx 0 \to {}^*f(y + \widehat{c}) \approx \widehat{l}. \quad (5.21)$$

When the outermost plan-specification is applied, the hypothesis fails to embed, so the embedding critic follows the plan shown in figure 5.2. This does not suggest a rule to rewrite the conclusion, which would allow the outermost plan-specification to continue. This is because the rewriting needs to take place in the hypotheses in order to yield a proof-plan. We have assumed in our implementation of lemma speculation via the embedding critic and the wave critic, that we need to find a way of rewriting the conclusion. We rewrite the hypotheses interactively and yield a proof-plan for the conjecture.

5.6 System Performance and results

We give a description of how the well the system performed on the both the development set and the test set. The theorems being tested in this chapter are relatively complex, and so we do not have a vast test set and development set. We perform some empirical studies which help us argue whether the plan-specifications and machinery presented in this chapter are successful at capturing the patterns of reasoning in the proofs.

5.6.1 Successes and Failures

The plan-specifications and associated methods for automating the construction of proof-plans described in this chapter were created by taking into account the structure of the proof observed in the development theorems. We show in tables 5.1 and 5.2 how well the system performed on each theorem. We show whether a complete proof-plan was yielded automatically, whether it had to be completed by specifying a new proof-plan interactively, or whether it was impossible to construct a complete proof-plan. We note also how lengthy the proof-plan for each theorem was in atomic method applications. This size refers to the plan construction of just the main goal in atomic method applications, and not the size of the plans for the lemmas which were speculated or needed for the theorem. We show how many critics fired, and how many lemmas were speculated automatically. We also show how many lemmas had to be introduced by hand. Finally we add a column to describe how many hours it took in development time to encode each theorem and achieve a complete proof-plan. For the development conjectures, this time indicates how long it took to achieve a complete proof-plan without the help of the plan-specifications we devised as a result of studying the development examples. For the test conjectures this represents the time taken to yield a complete proof-plan, or to give up attempting to construct one, given the planning machinery implemented as a result of the analysis of the development set. Those conjectures which have been automatically speculated during the proof-plan construction of the proof of another conjecture are denoted with a *.

The number of lemmas introduced by hand for the Product rule is 0, since $\lambda Clam$ was able to speculate the correct lemmas. In one case $\lambda Clam$ could not obtain a proof-plan for one lemma, but it was able to correctly speculate the correct lemma to use, so the value in this column is 0.

The test sets are relatively small and could have been augmented with more theorems such as l'Hôpital's rule and the Fundamental Theorem of Calculus. We did not have time to test the system on these theorems. These would have proved to be challenging theorems on account

Conjecture	Proof-plan yielded	Size of p-plan	No. critics fired	No. spec. lems	No. lems by hand	Dev. time /hours
LIM+	yes	8	2	2	0	12
LIM$\times$	yes	8	2	2	0	12
Chain Rule	yes	21	4	3	1	30
*Continuity of +	yes	4	0	0	0	24
*Continuity of $\times$	yes	4	0	0	0	12
*Rule (5.2)	yes	3	0	0	0	1
*Rule (5.10)	by hand	11	-	-	2	12
*Rule (5.12)	yes	4	0	0	0	1
*Rule (5.11)	yes	3	0	0	0	12

Table 5.1: Development set results

Conjecture	Proof-plan yielded	Size of p-plan	No. critics fired	No. spec. lems	No. lems by hand	Dev. time /hours
LIM /	yes	8	2	2	0	1
LIM $-$	yes	8	2	2	0	1
*Continuity of /	no	9	0	0	1	6
*Continuity of $-$	yes	3	0	0	0	1
Product rule	yes	11	3	3	0	36
Conjecture (5.21)	yes	5	1	1	2	8
*Rule (5.20)	yes	4	0	0	1	1

Table 5.2: Test set results

of their complexity. In particular we have not yet formally defined integration. Preliminary attempts at formalising integration can be seen in chapter 7.

5.6.2 Search space

In order to justify the use of the reasoning patterns set out in section 5.2.2, we must show that the proofs would not have been possible without use of these encoded heuristics, or at least the search space would have been very much bigger.

In order to discuss how the search space is constructed in our system we must indicate at which points backtracking can occur and show which groups of rules apply to which plans. Backtracking can occur in the following places:

- within the tautology method, which just applies logical rules;
- in the embedding critic for the substitution calculation;
- in rippling and rewriting choices can be made for which rule to apply.

The rules available to rippling are just those which can be annotated. These are rules (4.34) to (4.37) and (4.42) to (4.49). There is only one way in which these can be applied by rippling. When rules are suggested by lemma speculation, rippling can also annotate and use these rules. The planner prefers to fire a critic for the failure of rippling, but if the critic does not result in success, then symbolic evaluation is attempted. This has access to all of the rules, and hence can result in large search spaces.

As the system mentioned here uses critics to such a large extent, it is hard to make a judgement on the amount of search that is being performed internally. We perform three experiments to analyse the extent to which the search space has been reduced. In what follows, our analysis uses the chain rule as the motivating example.

Our system

In our plan-architecture, backtracking occurs in the fertilisation critic where a case-split is inserted at the initial node of the plan. This case-split is controlled by only employing it within the fertilisation critic. For the chain rule, the system explores 8 nodes of the plan before trying fertilisation, which fails, inducing backtracking to the initial node, where a case split is introduced. Also one other subgoal is introduced in order to allow fertilisation to take place. From this point, the evaluation methods complete the proof-plan. The number of atomic method applications for the resulting proof-plan is 29, indicating a negligible branching factor.

Naïve Strategy 1

We first attempted the chain rule with just access to symbolic evaluation using our axiomatisation, and an iterative deepening planner. We ran this experiment overnight, and reached a depth of 8, having explored roughly 100,000 nodes. At this point there were very many meta-variables introduced into the conclusion via transitivity and field rules. The average branching factor up to this point in the proof-plan is roughly 4.

Naïve Strategy 2

For the second experiment, we took the atomic methods, and their associated critics, and used an iterative deepening planner on a waterfall of these methods. In this case the planner explored roughly 100 nodes before completing a proof-plan for the chain rule, indicating a negligible branching factor.

Iterative Deepening with our plan-specification

In this experiment, we see that only 40 nodes are visited before a proof-plan for the chain rule is found. Again this constitutes a negligible branching factor.

5.6.3 Evaluation

We discuss the results shown in tables 5.1 and 5.2 with relation to the criteria set out in section 4.4. A similar evaluation scheme is given in [Cantu et al., 1996], where proof-planning was used to automate proof in large hardware verification problems.

Plan-specification criteria

- **Generality**

  In order to determine how well our plan-specifications can be reused, we must look at how easy it was to yield a complete proof-plan for the examples in the test set. It can be seen that for examples such as LIM $-$ that the mechanism works well; the time taken to develop the theorem is just that to encode it in $\lambda Clam$, and the necessary lemmas are speculated and planned automatically. LIM $/$ is more problematic; in order for a proof-plan to be yielded, we had to modify a speculated lemma by hand. Both the product rule and the extra limit conjecture (5.21) took some time, and required lemmas, which were added by hand. The time taken to yield proof-plans for these theorems is significant, and indicates that the plan-specifications are not general enough. Some of the speculated lemmas in this case are accounted for automatically by the evaluation plan-specifications, and are not trivial to automate.

  From the results of the search-space experiments we see that given the methods and plan-specifications we have provided, there is very little search. The most significant discrepancy in the search space size occurs where the axiomatisation is provided with

just an iterative deepening planner. In this case we see that our planning mechanisms reduce the search space considerably. This is because we encapsulate many applications of the axioms into each atomic method. For example, the evaluation methods rewrite terms in the conclusions using heuristics which make it simpler to show that the terms are infinitely close. These methods do automatically produce proof-plans for difficult parts of the proof, and do apply easily to problems from the test set. It must also be noted that a lot of work has to be done by hand in some cases to transform these plan-specifications to render them successful on the test examples.

- **Intuitiveness**

It is difficult to make a judgement as to whether the plan-specifications we develop correspond to a human intuition of how to carry out the proofs. Clearly the proof-plans do not correspond to the standard proofs presented in text books, but the general patterns are similar. In particular we claim that the three critics used, correspond to suggestions for proof-transformation which occur when attempting the proofs as a human. Reducing the form of conjecture to (5.14) through rules such as (4.54) mirrors the sort of reasoning which exists in the informal proofs of Newton and Leibniz. We have shown that this is possible in an automated setting, and hence that the resulting proof-plans mirror a form of reasoning which is believed to be intuitive.

- **Simplicity**

As can be seen from the description given in section 5.3, the plan-specifications are quite compact. The specifications can be so simple because a lot of the reasoning steps are encoded in the atomic methods. Also, critics further simplify plan-specifications since they analyse the failure of the associated methods and perform appropriate proof-transformations. This means that the main reasoning patterns which we wish to encompass are encapsulated in the methods, and the variations on these are described by the critic applications.

Process criteria

- **Prescriptiveness**

  Our planning machinery should greatly reduce the search space, in comparison with less informed searches. We can see from the experiments performed in section 5.6.2 that indeed the search space is greatly reduced. This is because the heuristics incorporated in the methods can plan large sections of each proof automatically.

- **Efficiency**

  As efficiency is not of direct interest to the idea of investigating the structure of proof, we do not make explicit timing measurements. The proof-plans in general take several minutes to complete. Instead, the size of the search space is viewed as more important. The parts of associated object level proofs which are computationally expensive are encoded mainly in the atomic methods which are guided by heuristics.

5.6.4 Comparison with other work

There has been little work done on automating the kinds of theorem presented in this chapter. [Bledsoe and Ballantyne, 1977] produced proofs of some of these theorems in a resolution theorem prover. They produced other proofs such as the Bolzano-Weierstraß conjecture, and their work was very successful. Their approach was different from ours as they were interested in yielding proofs without investigating the nature of the advantage given by non-standard analysis. In our work we produce proof-plans whose nodes describe various common patterns of reasoning which apply.

Some of the theorems presented here have been proved interactively in Isabelle, as described in [Fleuriot, 2001a]. The resulting proof-plans are very similar in shape to those that result from proving the theorems in Isabelle. This is partly because we chose to base our axiomatisation on the same extensional formalisation of non-standard analysis as that in Isabelle.

The ΩMEGA proof-planner [Benzmüller et al., 1997] has produced proof-plans for some of the limit theorems presented in this chapter. It uses standard analysis definitions to yield proof-plans, and is very successful at some types of conjecture. It bases its work on a sophisticated constraint solver, which calculates instantiations for the meta-variables introduced in standard proofs. It does not, however, have a critics mechanism such as ours, and cannot spec-

ulate lemmas during proof-plan execution as our system does. Comparing our development with that of the ΩMEGA proof-planner is difficult. Although the work described in this chapter studies some similar theorems to the Ω proof-planner (e.g. continuity of $\times$), the two systems are based on different mathematical theories. The ΩMEGA corpus is much larger than our set of examples, but the ones that we study are more fundamental and require more complicated proofs.

5.7 Discussion

We have analysed the performance of our system according to the evaluation criteria set out in section 4.4.2. There are some points worth noting about the types of proof-plan that are constructed for the theorems presented in this chapter. While it is the case that non-standard analysis provides us with a mathematical framework which allows the automation of proof for real analysis problems, we cannot claim that these proofs are more or less intuitive than their standard counterparts. The techniques we employ here are not complicated, and the problem of having to guess the instantiations of variables early in the proof is less than in the standard case.

The introduction of the $\approx$ relation through non-standard analysis hides a lot of complicated alternating quantifier terms from non-standard analysis. However, by introducing a new number system which includes infinite quantities, we cannot use some of the simplification procedures common to the usual reasoning over the reals without first ascertaining finiteness properties of the variables involved. This often results in finiteness conditions being imposed on the rules available to the system. Note also that when speculating lemmas it is easy to impose stringent conditions by applying rules naively. For example consider the continuity of times as stated by the lemma speculation machinery during the proof of LIM$\times$

$$a, b, c, d : {}^*\mathbb{R}$$
$$\mathcal{M}(a, b, c, d)$$
$$a \approx b$$
$$c \approx d \vdash$$
$$a \times c \approx b \times d.$$

If we attempt to use the rule

$$C \not\approx 0 \rightarrow A \times C \approx B \Rightarrow A \approx \frac{B}{C}$$

we end up with the condition that b and d in the conjecture cannot be infinitesimal. These conditions cannot be proved using just the hypotheses in the LIM$\times$ theorem. This is why we introduce rule (5.12) to mimic the substitution rule for equality.

After the work in this thesis had been completed it was noted that other possible definitions for derivative exists which may have simplified the proofs. In particular, Carathéodory's criterion for differentiability is of interest:

Theorem 7 *A function f is Carathéodory differentiable at a if there exists a function ϕ which is continuous at a such that*

$$f(x) - f(a) = \phi(x)(x-a)$$

It is not clear how using any other form of derivative would have rendered any proof simpler, as any definition for the derivative would still rely on the notion of limit, which produces the tricky extra cases that exist in, for example, the chain rule.

As mentioned in section 5.5.2, Method 4 is unsound and must be corrected. This was discovered after completion of the work, and has repercussions on the validity of any proof-plan which uses Method 4. Method 4 should be stated as in Method 9. The mistake we made is that we based the troublesome rewriting steps on axiom (4.53), but rewrote in the direction of the implication. The correct application of the axiom, written as a rewrite rule is

$$finite(Y) \rightarrow \quad X \times Y \approx Z \times Y \;\Rightarrow\; X \approx Z.$$

Thus if we want to multiply through by denominators, we state the rule as

$$finite(Y^{-1}) \rightarrow \quad X \times Y^{-1} \approx Z \times Y^{-1} \;\Rightarrow\; X \approx Z.$$

Thus from axiom (4.56) we deduce that the correct condition for multiplying through by a denominator is that the denominator is not infinitesimal.

The only two theorems affected by this are those already mentioned, as described in sections 5.5.2 and 5.5.4.

5.8 Summary

The research contribution of this work is to provide the proof-planning machinery by which proofs involving limits and associated concepts can be planned automatically. We have explored and implemented ideas from proof-planning which help us to encapsulate reasoning

Method 9 Correct version of multiplying through by denominators

Input: Goal: $\mathtt{H} \vdash L \approx R$

Conditions:

Exhaustively Apply rules:

$r \rightarrow, l \rightarrow$

$A \times (B + C) \rightarrow (A \times B) + (A \times C)$

$(A + B) \times C \rightarrow (A \times C) + (B \times C)$

Multiply R through by all denominators, D, of L, where $D \not\approx 0$

Multiply L through by all denominators, D, of R, where $D \not\approx 0$

Output: Goal: $\mathtt{H} \vdash L \approx R$

within a mathematical theory. We have shown that it is possible to construct plan-specifications which encapsulate the common patterns of reasoning in the proofs presented in this chapter.

In the next chapter, we enhance our framework by combining inductive arguments and non-standard analysis to deal with important theorems from real analysis.

Chapter 6

Incorporating induction

In this chapter we present the work done on automating the construction of plans for another area of analysis. In this case we study proofs of theorems which describe general properties of certain types of function. In particular we look at properties of uniformly continuous and differentiable functions over a closed interval. In order to reason about these functions we develop a technique which uses a recursive function to split the interval up. This allows us to conjecture properties about this recursive function, which we call the *partitioning function* and prove them by induction. Once the interval has been partitioned an infinite number of times, we can use non-standard analysis to analyse the function in an infinitesimal interval.

We first describe briefly the implementational research contribution of the work presented in this chapter. We then go on to present the main technique of this work using the Intermediate Value Theorem as an exemplary case study. Next we give detailed presentations of the plans constructed for Rolle's Theorem, which we believe to represent a novel proof. We then further present a version of the intermediate value theorem which follows ideas from constructive analysis, and mention the other theorems to which the technique has successfully been applied. We discuss the reasoning patterns which are evident in the proof-plans, and discuss the implementation done to encapsulate these patterns of reasoning. Finally we give an evaluation of the system, according to the criteria set out in chapter 4.

6.1 System enhancement

During the implementation of the work in this chapter, we included in $\lambda Clam$ the proof-plans and plan-specifications which were constructed, and also a more general way of dealing with

case-splits. Previously in λ*Clam* , a case-split was assumed to take place only if a conditional rewrite rule had a dual, in the sense that if the condition to one was X, then another similarly applicable rewrite rule had a condition which was $\neg X$. When a case split set is employed, the disjunctive composition of all of the conditions is set as a subgoal.

6.2 The technique

In this section, we discuss proofs, rather than plans, as we do not refer to any implementation. We describe the general form of the technique in section 6.2.4, but first illustrate it by means of an example. As our example we use the Intermediate value theorem, which stated in non-standard analysis is

$$f : \mathbb{R} \to \mathbb{R}$$

$$a, b, c : \mathbb{R}$$

$$\forall x, y \in {}^*\mathbb{R}.\ \widehat{a} \leq x \leq \widehat{b} \wedge \widehat{a} \leq y \leq \widehat{b} \wedge x \approx y \ \rightarrow\ {}^*f(x) \approx {}^*f(y) \tag{6.1}$$

$$a \leq b$$

$$f(a) \geq c \geq f(b)$$

$$\vdash \exists x \in \mathbb{R}.\ a \leq x \leq b \wedge f(x) = c. \tag{6.2}$$

It is important to notice that our characterisation of the theorem involves uniform continuity, which means that this is a slightly modified version of the standard statement for the Intermediate Value Theorem. Intuitively the theorem states that any uniformly continuous function attains all the values taken between the values at the end points of a closed interval. We approach the proof from an algorithmic perspective, reducing the size of the interval recursively, and showing that a point x satisfying $f(x) = c$ always lies within the interval. We illustrate this using figure 6.1. The intervals $[a_i, b_i]$ are decreasing with i, and always contain the point x. The interval $[a_{s(n)}, b_{s(n)}]$ is defined by assigning it to the left or right half of the interval $[a_n, b_n]$. As $f(a) > f(b)$ in the diagram, we choose the right half of the interval $[a_n, b_n]$, namely $[\frac{a_n+b_n}{2}, b_n]$, if $f(\frac{a_n+b_n}{2}) > c$, and the left half, namely $[a_n, \frac{a_n+b_n}{2}]$ otherwise. We refer to this choice of how to calculate successive partitions as the *partitioning criterion*. The idea of this approach is to show that an algorithm for finding a point with property specified by the conclusion of the Intermediate Value Theorem (6.2) will converge on that point at infinity.

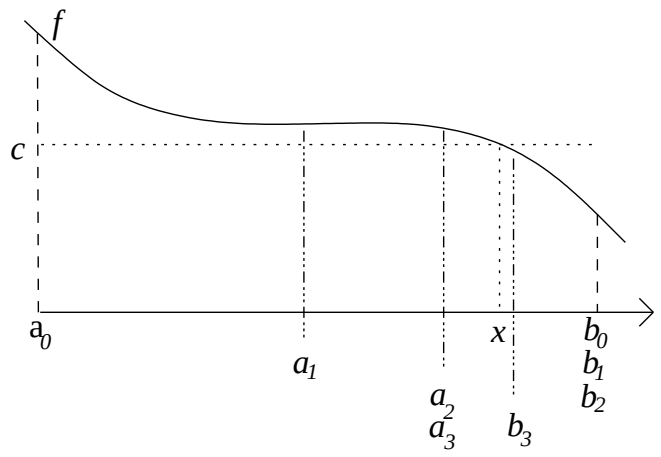

Figure 6.1: A sequence of partitions

```
ivtrec F A B C 0 = [A,B]
ivtrec F A B C s(N) =  (let [X,Y]=ivtrec F A B C N
                          in if F((X+Y)/2)>C then [(X+Y)/2,Y]
                                             else [X,(X+Y)/2])
```

Figure 6.2: The partitioning function for the Intermediate Value Theorem

6.2.1 Defining the *partitioning function*

We define a recursive function, which we henceforth refer to as the *partitioning function*, as shown in figure 6.2. This returns the intervals shown in figure 6.1 by $[a_i, b_i]$, and calculates successive partitions according to the partitioning criterion described above. We need to show that the interval returned for this function always contains a witness for x in the statement of the theorem. In order to do this we conjecture theorems about the partitioning function.

In λ*Clam* we represent the partitioning function by means of rewrite rules. Two sets of rules, $\texttt{ivtrec}_\texttt{l}$ and $\texttt{ivtrec}_\texttt{r}$, are attributed to the left and right end points of the interval respectively. For a full exposition of these rewrite rules, and how they can be annotated see figure 6.3, where we give the conditions to the rewrite rules in red for clarity. We can now state conjectures about the partitioning function, and use the wave rules to produce proof-plans.

$$\texttt{ivtrec}_\texttt{l}\ F\ A\ B\ C\ 0 \Rightarrow A$$

$$\texttt{ivtrec}_\texttt{r}\ F\ A\ B\ C\ 0 \Rightarrow B$$

$$F((\texttt{ivtrec}_\texttt{l}\ F\ A\ B\ C\ N + \texttt{ivtrec}_\texttt{r}\ F\ A\ B\ C\ N)/2) \leq C \rightarrow$$
$$\texttt{ivtrec}_\texttt{l} F\ A\ B\ C\ \boxed{s(N)}^{\uparrow} \Rightarrow \texttt{ivtrec}_\texttt{l}\ F\ A\ B\ C\ N$$

$$F((\texttt{ivtrec}_\texttt{l}\ F\ A\ B\ C\ N + \texttt{ivtrec}_\texttt{r}\ F\ A\ B\ C\ N)/2) \leq C \rightarrow$$
$$\texttt{ivtrec}_\texttt{r} F\ A\ B\ C\ \boxed{s(N)}^{\uparrow} \Rightarrow \boxed{(\texttt{ivtrec}_\texttt{r}\ F\ A\ B\ C\ N + \texttt{ivtrec}_\texttt{l}\ F\ A\ B\ C\ N)/2}^{\uparrow}$$

$$F((\texttt{ivtrec}_\texttt{l}\ F\ A\ B\ C\ N + \texttt{ivtrec}_\texttt{r}\ F\ A\ B\ C\ N)/2) > C \rightarrow$$
$$\texttt{ivtrec}_\texttt{l} F\ A\ B\ C\ \boxed{s(N)}^{\uparrow} \Rightarrow \boxed{(\texttt{ivtrec}_\texttt{l}\ F\ A\ B\ C\ N + \texttt{ivtrec}_\texttt{r}\ F\ A\ B\ C\ N)/2}^{\uparrow}$$

$$F((\texttt{ivtrec}_\texttt{l}\ F\ A\ B\ C\ N + \texttt{ivtrec}_\texttt{r}\ F\ A\ B\ C\ N)/2) > C \rightarrow$$
$$\texttt{ivtrec}_\texttt{r} F\ A\ B\ C\ \boxed{s(N)}^{\uparrow} \Rightarrow \texttt{ivtrec}_\texttt{r}\ F\ A\ B\ C\ N$$

Figure 6.3: The wave rules representing the partitioning function for the Intermediate Value Theorem

6.2.2 Abbreviated definitions for inductive theorems

We state various lemmas about the partitioning function to be

$$\forall n \in \mathbb{N}.\ \texttt{ivtrec}_\texttt{r}\ f\ a\ b\ c\ n - \texttt{ivtrec}_\texttt{l}\ f\ a\ b\ c\ n = \tfrac{b-a}{2^n} \tag{6.3}$$

$$\forall n \in \mathbb{N}.\ \texttt{ivtrec}_\texttt{r}\ f\ a\ b\ c\ n \geq a \wedge \forall n \in \mathbb{N}.\ \texttt{ivtrec}_\texttt{r}\ f\ a\ b\ c\ n \leq b \tag{6.4}$$

$$\forall n \in \mathbb{N}.\ \texttt{ivtrec}_\texttt{l}\ f\ a\ b\ c\ n \geq a \wedge \forall n \in \mathbb{N}.\ \texttt{ivtrec}_\texttt{l}\ f\ a\ b\ c\ n \leq b \tag{6.5}$$

$$\forall n \in \mathbb{N}.\ f(\texttt{ivtrec}_\texttt{r}\ f\ a\ b\ c\ n) \geq c \wedge c \geq f(\texttt{ivtrec}_\texttt{l}\ f\ a\ b\ c\ n). \tag{6.6}$$

These lemmas about the partitioning function can be proved inductively. Once proved, they can be transferred to the non-standard domain. From there we can reason about their properties at infinite hypernaturals.

We abbreviate each of the lemmas by attributing new variables to the common subterms. Let $l(n) = \texttt{ivtrec}_\texttt{l}\ f\ a\ b\ c\ n$, and $r(n) = \texttt{ivtrec}_\texttt{r}\ f\ a\ b\ c\ n$. We can then transfer to the

non-standard domain yielding for example

$$\forall n \in {}^*\mathbb{N}.\ {}^*r(n) - {}^*l(n) = \frac{\widehat{b} - \widehat{a}}{2^n}.$$

Using the abbreviated and transferred forms of (6.3)–(6.6) we can write a number of statements about the transferred terms ${}^*l(n)$ and ${}^*r(n)$ for an infinite hypernatural n. In particular, using the additional fact that $\frac{\widehat{b}-\widehat{a}}{2^n} \approx 0$:

$$\begin{gathered}
{}^*l(n) \approx {}^*r(n) \\
\widehat{a} \leq {}^*r(n) \leq \widehat{b} \\
\widehat{a} \leq {}^*l(n) \leq \widehat{b} \\
{}^*f({}^*l(n)) \geq \widehat{c} \geq {}^*f({}^*r(n))
\end{gathered}$$

6.2.3 Using non-standard analysis

Using the abbreviated inductive theorems we can reformulate the Intermediate Value Theorem as follows:

$$\begin{gathered}
l, r : {}^*\mathbb{N} \to {}^*\mathbb{R} \\
n : {}^*\mathbb{N} \\
f : \mathbb{R} \to \mathbb{R} \\
a, b, c : \mathbb{R} \\
\neg \mathit{finite}(n) \\
{}^*l(n) \approx {}^*r(n) \\
\widehat{a} \leq {}^*r(n) \leq \widehat{b} \\
\widehat{a} \leq {}^*l(n) \leq \widehat{b} \\
{}^*f({}^*l(n)) \geq \widehat{c} \geq {}^*f({}^*r(n)) \\
\forall x, y \in {}^*\mathbb{R}.\ \widehat{a} \leq x \leq \widehat{b} \wedge \widehat{a} \leq y \leq \widehat{b} \wedge x \approx y \ \to\ {}^*f(x) \approx {}^*f(y) \\
\vdash \exists x \in \mathbb{R}.\ a \leq x \leq b \wedge f(x) = c
\end{gathered}$$

Now we can use techniques from non-standard analysis to complete the proof. We use $\lambda Clam$ to manipulate the knowledge in the hypotheses using forward reasoning steps. For example, we can use uniform continuity to deduce more properties about ${}^*l(n)$ and ${}^*r(n)$ such as

$${}^*f({}^*l(n)) \approx {}^*f({}^*r(n)).$$

Now it follows that ${}^*f({}^*l(n)) \approx \widehat{c}$, since ${}^*f({}^*l(n)) \geq \widehat{c} \geq {}^*f({}^*r(n))$. From continuity it also follows that

$$\exists x \in \mathbb{R}.\ \widehat{a} \leq \widehat{x} \leq \widehat{b} \wedge {}^*l(n) \approx \widehat{c} \wedge {}^*f({}^*l(n)) \approx \widehat{f(x)}.$$

From transitivity

$$\widehat{f(x)} \approx \widehat{c}$$

from which it follows that

$$\exists x \in \mathbb{R}.\ a \leq x \leq b \wedge f(x) = c$$

as required.

6.2.4 General overview of technique

Now that a brief presentation of an example proof has been given, we can describe in more general terms the structure we have identified for this type of proof. Figure 6.4 shows the results of our investigations and is described next.

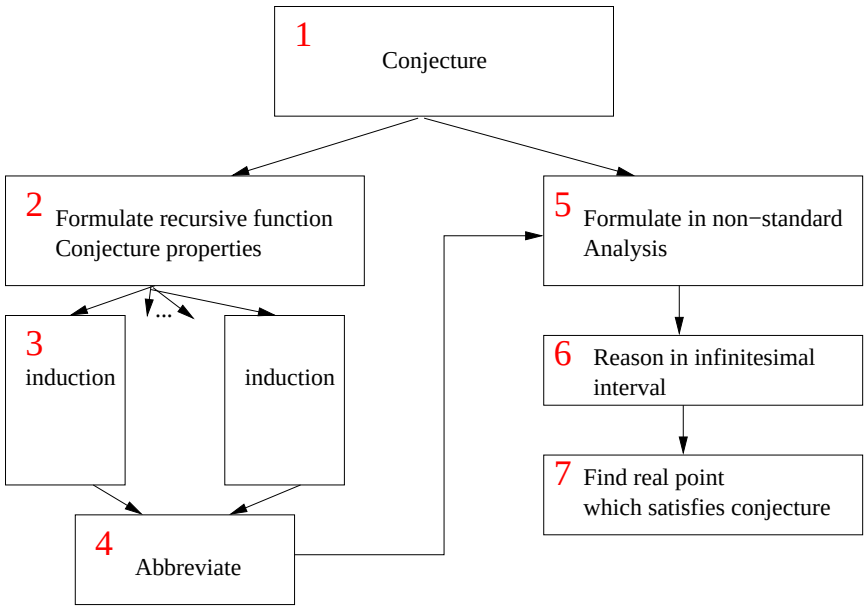

Figure 6.4: Proof architecture

1. Conjecture

We state the conjecture using the non-standard characterisations for limit, continuity and derivative. We add subsequent results derived from the conjectures about the partitioning function to the hypotheses.

2. Formulate partitioning function

The partitioning function is defined. Its precise behaviour is determined by the partitioning criterion, but many of the inductive lemmas are independent of the this and the proof-plans are identical modulo the name and number of arguments of the partitioning function.

3. Prove by induction properties of partitioning function

We state by hand various lemmas about the partitioning function. These properties can be proved using structural induction on the natural numbers. $\lambda Clam$ uses rippling, together with the other powerful techniques which exist to tackle inductive conjectures in $\lambda Clam$.

4. Abbreviate

Once the proofs of the conjectures about the partitioning function have been planned, the results are abbreviated by $\lambda Clam$, leaving them in the form $\forall n \in \mathbb{N}.\ P(n)$. This is done automatically, as described in section 6.5.

5. Transfer to hypernaturals and hyperreals

$\lambda Clam$ transfers the abbreviated conjectures to the non-standard domain using the transfer principle set out in section 2.4.1. It then reasons about the abbreviated conjectures at an infinite hypernatural, deducing properties of the resulting infinitesimal partition. This is done automatically– see section 6.5 for a description of the procedure.

6. Reason using non-standard analysis

The proofs we perform using the abbreviated inductive conjectures are all forward reasoning proofs. $\lambda Clam$ rewrites the hypotheses to yield the result we need. This is because we introduce extra existential variables into the hypotheses and then deduce properties about them using appropriate definitions, such as that for uniform continuity for example.

7. Find real point that satisfies conjecture

$\lambda Clam$ generates a real number witness finally for the real analysis result by appealing to rule (4.40) which allows us to remove the non-standard annotations and replace $\approx$ with $=$, and hence move back into the real domain making use of rule (4.40).

6.3 The development set

We include only two real analysis theorems in this set of conjectures, as they are comprised of many user-stated and automatically speculated lemmas, and complicated proof architectures. In this section we present in detail the steps given by the proof-plan found for Rolle's Theorem. The presentation gives a proof which is guided by the steps specified by the proof-plan. A similar presentation is given for Intermediate Value Theorem in appendix B.1.

This section contains the proof-plan for a proof of Rolle's Theorem which we believe to be novel. We use definitions for *uniform* continuity and *uniform* differentiability as these are more natural for reasoning in non-standard analysis- as discussed in [Hoskins, 1990]. During the presentation we point out common reasoning techniques in bold font which we later refer to in section 6.4.

6.3.1 Rolle's Theorem

Figure 6.5 show our statement of Rolle's Theorem for real analysis using non-standard definitions for continuity (6.7) and uniform differentiability (6.8). It should be noted that the conclusion for this proof is slightly different from the standard version, as we are allowing the range of the existential variable to include the end points of the interval. Also we require f' to be uniformly differentiable in the closed interval $[a, b]$, so that we can determine the sign of the derivative at the end point of the interval. For a discussion and justification of this see section 6.9.

As with the Intermediate Value Theorem, we introduce a recursive function for which we can encode a set of wave rules to prove inductive conjectures that we state. In this case we introduce a more complicated partitioning criterion which ensures that there is a point with zero derivative in any interval. We do this by considering the sign of derivative and the relative positions of the endpoints and their midpoint. Figure 6.6 shows our construction of the partitioning criterion for Rolle's Theorem. Because we begin where the endpoints are equal, we can guarantee that we are always dealing with one of the cases shown in figure 6.6 at any step of the partitioning

$$f', f : \mathbb{R} \to \mathbb{R}$$

$$a, b : \mathbb{R}$$

$$\forall x, y \in {}^*\mathbb{R}.\ \widehat{a} \leq x \leq \widehat{b} \wedge \widehat{a} \leq y \leq \widehat{b} \wedge x \approx y \ \rightarrow\ {}^*f(x) \approx {}^*f(y) \tag{6.7}$$

$$\forall x \in {}^*\mathbb{R}.\ \forall h \in {}^*\mathbb{R}.\ \widehat{a} \leq x \leq \widehat{b} \wedge h \approx 0 \wedge h \neq 0 \ \rightarrow \frac{{}^*f(x+h) - {}^*f(x)}{h} \approx {}^*f'(x) \tag{6.8}$$

$$a < b$$

$$f(a) = f(b)$$

$$\vdash\ \exists x \in \mathbb{R}.\ a \leq x \leq b \wedge f'(x) = 0$$

Figure 6.5: Our characterisation of Rolle's Theorem

function.

We represent the algorithm using the partitioning function shown in figure 6.8. This function is then expressed in $\lambda Clam$ by the set of wave rules about the end-points of the interval given in figure 6.9, where the conditions are shown in red for clarity.

Next $\lambda Clam$ conjectures the following lemmas about the partitioning function:

$$\forall n \in \mathbb{N}.\ \mathtt{rolrec_r}\ f\ f'\ a\ b\ n\ -\ \mathtt{rolrec_l}\ f\ f'\ a\ b\ n = \frac{b-a}{2^n} \tag{6.9}$$

$$\forall n \in \mathbb{N}.\ \mathtt{rolrec_r}\ f\ f'\ a\ b\ n\ \leq\ b \wedge \mathtt{rolrec_r}\ f\ f'\ a\ b\ n\ \geq a \tag{6.10}$$

$$\forall n \in \mathbb{N}.\ \mathtt{rolrec_l}\ f\ f'\ a\ b\ n\ \leq\ b \wedge \mathtt{rolrec_l}\ f\ f'\ a\ b\ n\ \geq a \tag{6.11}$$

$$\forall n \in \mathbb{N}.\ \mathtt{rolrec_r}\ f\ f'\ a\ b\ n\ >\ \mathtt{rolrec_l}\ f\ f'\ a\ b\ n \tag{6.12}$$

The more interesting conjecture is one specific to the partitioning criterion for Rolle's Theorem shown in figure 6.8. In particular, we want to show that the end points of any interval obey one of the cases in figure 6.10, so we conjecture the lemma

$$\begin{aligned}
&\forall n \in \mathbb{N}.\\
&\quad f'(\mathtt{rolrec_l}\ f\ f'\ a\ b\ n) \geq 0 \ \wedge\ f'(\mathtt{rolrec_r}\ f\ f'\ a\ b\ n) < 0 \ \vee\\
&\quad f'(\mathtt{rolrec_l}\ f\ f'\ a\ b\ n) < 0 \ \wedge\ f'(\mathtt{rolrec_r}\ f\ f'\ a\ b\ n) \geq 0 \ \vee\\
&\quad f'(\mathtt{rolrec_l}\ f\ f'\ a\ b\ n) \geq 0 \ \wedge\ f'(\mathtt{rolrec_r}\ f\ f'\ a\ b\ n) \geq 0\\
&\qquad\qquad \wedge\ f(\mathtt{rolrec_l}\ f\ f'\ a\ b\ n) \geq f(\mathtt{rolrec_r}\ f\ f'\ a\ b\ n) \ \vee\\
&\quad f'(\mathtt{rolrec_l}\ f\ f'\ a\ b\ n) < 0 \ \wedge\ f'(\mathtt{rolrec_r}\ f\ f'\ a\ b\ n) < 0\\
&\qquad\qquad \wedge\ f(\mathtt{rolrec_l}\ f\ f'\ a\ b\ n) \leq f(\mathtt{rolrec_r}\ f\ f'\ a\ b\ n).
\end{aligned} \tag{6.13}$$

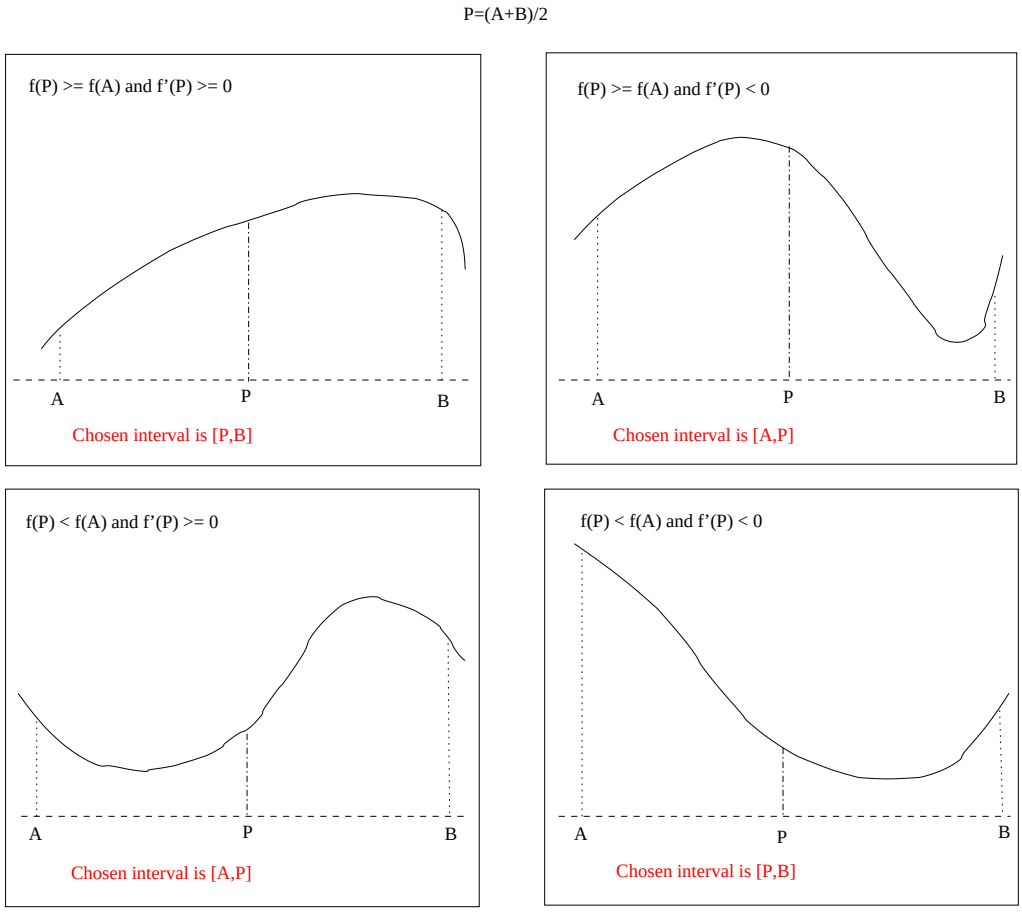

Figure 6.6: The cases that constitute the partitioning criterion for Rolle's Theorem

Setting up the reformulation of Rolle's Theorem

We show in this section how $\lambda Clam$ abbreviates conjectures (6.9)–(6.13) and adds them to the hypotheses of the original goal about Rolle's Theorem. Following the plan-specification which will be presented in more details in section 6.5, we introduce two new functions r and l, which we assign as $l(n) = \texttt{rolrec}_{\texttt{l}}\ f\ f'\ a\ b\ n$ and $r(n) = \texttt{rolrec}_{\texttt{r}}\ f\ f'\ a\ b\ n$. Now the inductive theorems can be reformulated and transferred to the non-standard domain giving:

$$\forall n \in {}^*\mathbb{N}.\ {}^*r(n) - {}^*l(n) = \frac{\widehat{b} - \widehat{a}}{2^n}$$
$$\forall n \in {}^*\mathbb{N}.\ {}^*r(n) > {}^*l(n)$$
$$\forall n \in {}^*\mathbb{N}.\ \widehat{a} \leq {}^*r(n) \leq \widehat{b}$$
$$\forall n \in {}^*\mathbb{N}.\ \widehat{a} \leq {}^*l(n) \leq \widehat{b}$$

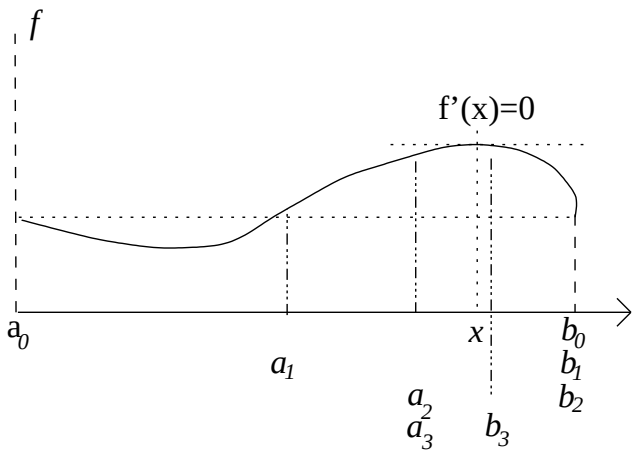

Figure 6.7: Finding a point of zero derivative

$$\begin{aligned}\forall n \in {}^*\mathbb{N}. \quad & ({}^*f'({}^*l(n)) \geq 0 \wedge {}^*f'({}^*r(n)) < 0) \ \vee \\ & ({}^*f'({}^*l(n)) < 0 \wedge {}^*f'({}^*r(n)) < 0 \wedge {}^*f({}^*l(n)) \leq {}^*f({}^*r(n))) \vee \\ & ({}^*f'({}^*l(n)) < 0 \wedge {}^*f'({}^*r(n)) \geq 0) \vee \\ & ({}^*f'({}^*l(n)) \geq 0 \wedge {}^*f'({}^*r(n)) \geq 0 \wedge {}^*f({}^*l(n)) \geq {}^*f({}^*r(n)))\end{aligned}$$

Next, we apply the plan-specification set out later in section 6.5.5 for converting these formulae into a simple form which can be added to the hypotheses of Rolle's Theorem. Using the plan-specification, this allows us to reformulate Rolle's Theorem with the new hypotheses

$$l, r : {}^*\mathbb{N} \to {}^*\mathbb{R}$$

$$n : {}^*\mathbb{N}$$

$$\neg finite(n)$$

$${}^*r(n) \approx {}^*l(n)$$

$$\widehat{a} \leq {}^*l(n) \leq \widehat{b}$$

$$\widehat{a} \leq {}^*r(n) \leq \widehat{b}$$

$${}^*r(n) > {}^*l(n)$$

$${}^*f({}^*r(n)) \approx {}^*f({}^*l(n))$$

$${}^*f'({}^*r(n)) \approx {}^*f'({}^*l(n))$$

$$({}^*f'({}^*l(n)) \geq 0 \wedge {}^*f'({}^*r(n)) < 0) \vee ({}^*f'({}^*l(n)) < 0 \wedge {}^*f'({}^*r(n)) < 0 \wedge {}^*f({}^*l(n)) \leq {}^*f({}^*r(n))) \vee$$

$$({}^*f'({}^*l(n)) < 0 \wedge {}^*f'({}^*r(n)) \geq 0) \vee ({}^*f'({}^*l(n)) \geq 0 \wedge {}^*f'({}^*r(n)) \geq 0 \wedge {}^*f({}^*l(n)) \geq {}^*f({}^*r(n))).$$

```
rolrec F F' A B 0 = [A,B]

rolrec F F' A B s(N) =
      (let [X,Y]=rolrec F F' A B N
         in
               if F((X+Y)/2) >= F(X) and F'((X+Y)/2) >= 0
                           then [(X+Y)/2,Y]

           else if F((X+Y)/2) >= F(X) and F'((X+Y)/2) < 0
                           then [X,(X+Y)/2]

           else if F((X+Y)/2) < F(X) and F'((X+Y)/2) >= 0
                           then [X,(X+Y)/2]

           else if F((X+Y)/2) < F(X) and F'((X+Y)/2) < 0
                           then [(X+Y)/2,Y])
```

Figure 6.8: The partitioning function for Rolle's Theorem

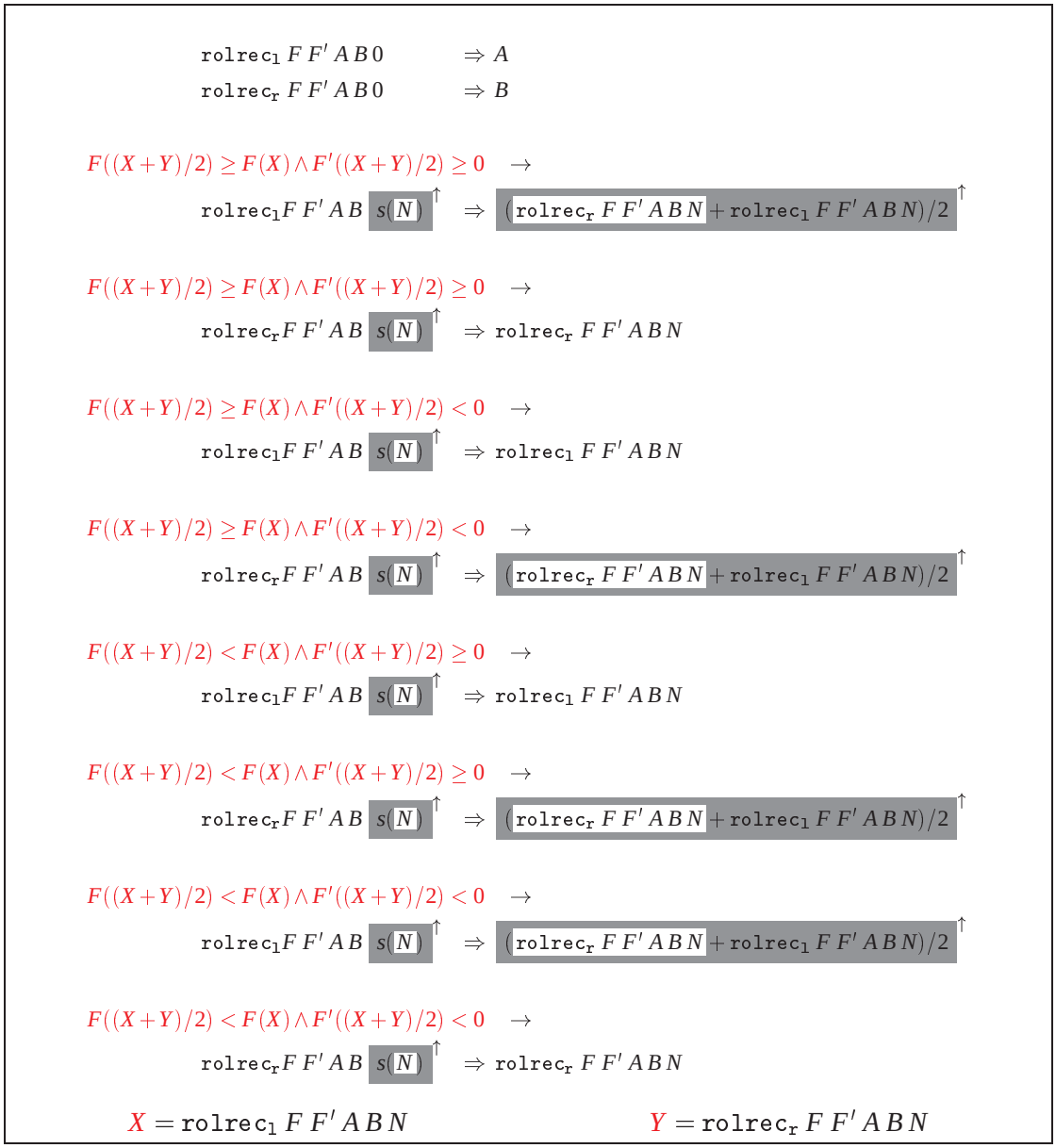

Figure 6.9: The wave rules representing the partitioning function for Rolle's Theorem

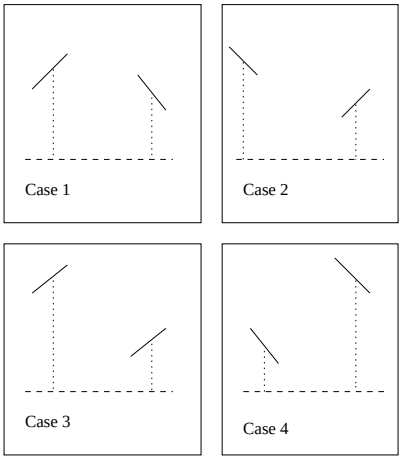

Figure 6.10: The four possible situations of the end points of the partition for Rolle's theorem

Note that $\lambda Clam$ appeals to uniform continuity to yield ${}^*f({}^*l(n)) \approx {}^*f({}^*r(n))$ from ${}^*r(n) \approx {}^*l(n)$, and uses the following lemma

$$\forall x, y \in {}^*\mathbb{R}.\ x \approx y \rightarrow {}^*f'(x) \approx {}^*f'(y) \tag{6.14}$$

to obtain ${}^*f'({}^*l(n)) \approx {}^*f'({}^*r(n))$. To see why it is possible for us to this lemma, which states the uniform continuity of f' see section 6.3.2. Deriving these facts and adding them to the hypotheses is an example of a reasoning pattern, which we call **discharging conditions**.

Proving Rolle's Theorem using non-standard analysis

Here we describe how non-standard analysis is used to manipulate the information in the hypotheses to yield a witness for the conclusion. We describe the steps of the proof-plan yielded for Rolle's Theorem. We use axiom (4.39) to introduce a real variable to the hypotheses. Recall that this axiom is stated as

$$\forall X : {}^*\mathbb{R}.\ finite(X) \rightarrow \exists Y : \mathbb{R}.\ X \approx \widehat{Y}$$

We know that ${}^*l(n)$ is finite since the hypothesis holds

$$\widehat{a} \leq {}^*l(n) \leq \widehat{b}$$

so, we know that introducing a new real variable x to the hypotheses, and establishing that $\widehat{x} \approx {}^*l(n)$ is a valid proof step. Using this reasoning, $\lambda Clam$ employs existential elimination (rule $l\exists$) from our sequent calculus, and adds two facts to the hypotheses automatically:

$$
\begin{gathered}
l, r : {}^*\mathbb{N} \to {}^*\mathbb{R} \\
n : {}^*\mathbb{N} \\
f', f : \mathbb{R} \to \mathbb{R} \\
a, b : \mathbb{R} \\
\neg\, finite(n) \\
{}^*l(n), {}^*r(n) : {}^*\mathbb{R} \\
{}^*r(n) \approx {}^*l(n) \qquad (6.15) \\
\widehat{a} \leq {}^*l(n) \leq \widehat{b} \\
\widehat{a} \leq {}^*r(n) \leq \widehat{b} \\
{}^*r(n) > {}^*l(n) \\
{}^*f({}^*r(n)) \approx {}^*f({}^*l(n)) \\
{}^*f'({}^*r(n)) \approx {}^*f'({}^*l(n)) \\
({}^*f'({}^*l(n)) \geq 0 \wedge {}^*f'({}^*r(n)) < 0) \vee ({}^*f'({}^*l(n)) < 0 \wedge {}^*f'({}^*r(n)) < 0 \wedge {}^*f({}^*l(n)) \leq {}^*f({}^*r(n))) \vee \\
({}^*f'({}^*l(n)) < 0 \wedge {}^*f'({}^*r(n)) \geq 0) \vee ({}^*f'({}^*l(n)) \geq 0 \wedge {}^*f'({}^*r(n)) \geq 0 \wedge {}^*f({}^*l(n)) \geq {}^*f({}^*r(n))) \qquad (6.16) \\
\forall x, y \in {}^*\mathbb{R}.\ \widehat{a} \leq x \leq \widehat{b} \wedge \widehat{a} \leq y \leq \widehat{b} \wedge x \approx y \ \to\ {}^*f(x) \approx {}^*f(y) \\
\forall x \in {}^*\mathbb{R}.\ \forall h \in {}^*\mathbb{R}.\ a \leq x \leq b \wedge h \approx 0 \wedge h \neq 0 \ \to \frac{{}^*f(x+h) - {}^*f(x)}{h} \approx {}^*f'(x) \\
a < b \\
f(a) = f(b) \\
\vdash\ \exists x \in \mathbb{R}.\ a \leq x \leq b \wedge f'(x) = 0
\end{gathered}
$$

Figure 6.11: The reformulation of Rolle's Theorem

- $\widehat{x} \approx {}^*l(n)$

  This ensures the existence of a real number in the infinitesimal neighbourhood of any interval in the sequence produced by the partitioning function. This is an example of **witness introduction**.

- $a \leq x \leq b$

  $\lambda Clam$ establishes bounds on the introduced real variable x, using the facts that $\widehat{x} \approx {}^*l(n)$

and $\widehat{a} \leq {}^*l(n) \leq \widehat{b}$. In order to do this we introduce the following lemmas interactively:

$$A \approx B \wedge A > X \rightarrow B \approx X \vee B > X$$
$$B \approx A \wedge A > X \rightarrow B \approx X \vee B > X$$
$$A \approx B \wedge A < X \rightarrow B \approx X \vee B < X$$
$$B \approx A \wedge A < X \rightarrow B \approx X \vee B < X.$$

Using rules in this way to establish bounds on the variable x is an example of using **order constraints**.

$\lambda Clam$ now introduces four cases to the proof of Rolle's Theorem, each corresponding to a disjunct in hypothesis (6.16). For each case we show how $\lambda Clam$ yields a proof-plan for lemma ${}^*f'({}^*l(n)) \approx 0$, and hence establishes a witness for the conclusion of Rolle's Theorem. In each case n represents an infinite hypernatural. Each bullet point represents a goal which is proved using the argument in the corresponding text.

Case 1: ${}^*\mathtt{f}'({}^*\mathtt{l}(\mathtt{n})) \geq \mathtt{0} \wedge {}^*\mathtt{f}'({}^*\mathtt{r}(\mathtt{n})) < \mathtt{0}$

- ${}^*f'({}^*l(n)) \approx 0$

  We introduce the following rules interactively to $\lambda Clam$:

$$\begin{aligned} A \approx B \wedge A < 0 \wedge B > 0 &\rightarrow A \approx 0 \\ A \approx B \wedge A < 0 \wedge B > 0 &\rightarrow B \approx 0. \end{aligned} \tag{6.17}$$

  We know that ${}^*f'({}^*l(n)) \geq 0 \wedge {}^*f'({}^*r(n)) < 0$. We perform a case split on ${}^*f'({}^*l(n)) > 0 \vee {}^*f'({}^*l(n)) = 0$. In the first case we use rule (6.17) to yield ${}^*f'({}^*l(n)) \approx 0$ as required. In the second case we use rule (4.31) to establish the result. The reasoning involved here is another instance of using **order constraints**.

- $f'(x) = 0$

  Now we have the fact that ${}^*f'({}^*l(n)) \approx 0$ and we can use lemma (6.14) to ascertain that

$$\widehat{f'(x)} \approx {}^*f'({}^*l(n))$$

  and hence by transitivity of $\approx$ we can write

$$\widehat{f'(x)} \approx 0,$$

and hence by (4.40), $f'(x) = 0$.

Case 2: ${}^*\mathtt{f}'({}^*\mathtt{l}(\mathtt{n})) < \mathtt{0} \wedge {}^*\mathtt{f}'({}^*\mathtt{r}(\mathtt{n})) \geq \mathtt{0}$

We yield a proof-plan for $f'(x) = 0$ here as well. The steps of the proof-plan are omitted as they follow very closely the steps used in the case where ${}^*\mathtt{f}'({}^*\mathtt{l}(\mathtt{n})) \geq \mathtt{0} \wedge {}^*\mathtt{f}'({}^*\mathtt{r}(\mathtt{n})) < \mathtt{0}$.

Case 3: ${}^*\mathtt{f}'({}^*\mathtt{l}(\mathtt{n})) < \mathtt{0} \wedge {}^*\mathtt{f}'({}^*\mathtt{r}(\mathtt{n})) < \mathtt{0} \wedge {}^*\mathtt{f}({}^*\mathtt{l}(\mathtt{n})) \leq {}^*\mathtt{f}({}^*\mathtt{r}(\mathtt{n}))$

For this case we need to do more complicated reasoning about the sign of the derivative at each of the points ${}^*l(n)$ and ${}^*r(n)$. In order to do this we require the uniformly differentiability condition we imposed on f.

- ${}^*f'({}^*l(n)) \approx 0$

  Using the information in the hypotheses about ${}^*r(n)$ and ${}^*l(n)$ $\lambda Clam$ can use **simplification of derivatives** to yield

  $$ {}^*f'({}^*l(n)) \approx \frac{{}^*f({}^*r(n)) - {}^*f({}^*l(n))}{{}^*r(n) - {}^*l(n)}. \tag{6.18} $$

  We now consider two cases: one where ${}^*f({}^*l(n)) = {}^*f({}^*r(n))$ and one where ${}^*f({}^*l(n)) < {}^*f({}^*r(n))$.

  In the first case, we can evaluate the fraction given in (6.18) using the rules

  $$ A = B \rightarrow A - B = 0 \tag{6.19} $$

  $$ A \neq 0 \rightarrow \frac{0}{A} = 0 \tag{6.20} $$

  yielding the result ${}^*f'({}^*l(n)) \approx 0$ as required, since we know that ${}^*r(n) - {}^*l(n) \neq 0$.

  In the second case, we analyse expression (6.18), and use the rules

  $$ A > 0 \wedge B > 0 \rightarrow \frac{A}{B} > 0 \tag{6.21} $$

  $$ A > 0 \wedge B \approx A \rightarrow B \approx 0 \vee B > 0 \tag{6.22} $$

  together with rule 6.19 to determine that

  $$ {}^*f'({}^*l(n)) \approx 0 \ \vee \ {}^*f'({}^*l(n)) > 0. $$

The first disjunct is easily discharged. For the second case a contradiction is yielded since ${}^*f'({}^*l(n)) < 0$ in this branch of the case split, and the proof-plan is completed using rule *f_axiom* our sequent calculus.

In this case of the proof, the reasoning patterns we observe are examples of **order constraints** and **simplification of derivatives**.

- $f'(x) = 0$
  We prove this in the same way as the previous cases.

Case 4: ${}^*\mathtt{f}'({}^*\mathtt{l}(\mathtt{n})) \geq 0 \wedge {}^*\mathtt{f}'({}^*\mathtt{r}(\mathtt{n})) \geq 0 \wedge {}^*\mathtt{f}({}^*\mathtt{l}(\mathtt{n})) \geq {}^*\mathtt{f}({}^*\mathtt{r}(\mathtt{n}))$

The reasoning involved for this case follows exactly the shape of the proof-plan for the previous case, and so we omit the details.

Inductive lemmas

It now remains for us to prove the inductive lemmas (6.9),(6.10),(6.11),(6.12) and (6.13), which allowed us to add the abbreviated hypotheses to the statement of Rolle's Theorem. Presentations of the proof-plans yielded by the induction plan-specification for these inductive lemmas are given in section B.2. In all of the presentations of the proofs we use **case analysis** and **rippling**. We refer to lemmas (6.9), (6.10),(6.11) and (6.12) as the **common inductive lemmas**.

As an example of the reasoning patterns observed in these proof-plans, we consider the proof-plans for lemmas (6.10) and (6.11). For simplicity we write

$$r(n) = \mathtt{rolrec}_\mathtt{r}\ f\ f'\ a\ b\ n$$
$$l(n) = \mathtt{rolrec}_\mathtt{l}\ f\ f'\ a\ b\ n$$

We can then write these lemmas together as a conjunct:

$$\forall n \in \mathbb{N}.\ r(n)\ \leq\ b \wedge r(n)\ \geq a\ \wedge\ l(n)\ \leq\ b \wedge l(n)\ \geq a.$$

In order to yield proof-plans for these goals, we must use results from the proof of one conjecture in order to prove the other. In $\lambda Clam$, we simulate a mutual induction scheme. We perform induction the conjunction of all the goals, and then we can use the induction hypothesis of any of the conjuncts in order to rewrite the conclusion.

We study the step case here in order to point out the patterns of reasoning observed. The induction hypotheses are

$$n : \mathbb{N}$$
$$r(n) \leq b$$
$$r(n) \geq a$$
$$l(n) \leq b$$
$$l(n) \geq a$$

Consider one conjunct from the conclusion:

$$r(\,s(n)^{\uparrow}\,) \leq b.$$

When the case-split set (see figure 6.9) is applied, in one case this conclusion is rewritten to

$$\left(\frac{l(n)+r(n)}{2}\right)^{\uparrow} \leq b.$$

We refer to the reasoning pattern of using the case-split set in this way as **case analysis**. Notice here that the induction hypothesis from the conjunct $l(n) \leq b$ is embedded in the conclusion. This is an example of both **mutual induction** and **coloured rippling**. In order to complete the proof-plan, some lemmas are needed, as can be seen in section 6.6.2.

6.3.2 Lemma 6.14: uniform differentiability lemma

To obtain a proof-plan for (6.14) we need the definition for uniform differentiability, and using it, we state the lemma

$$f, f' : \mathbb{R}$$
$$\forall x \in {}^*\mathbb{R}.\ \forall h \in {}^*\mathbb{R}.\ \ h \approx 0 \wedge h \neq 0 \ \rightarrow \frac{{}^*f(x+h) - {}^*f(x)}{h} \approx {}^*f'(x) \quad \vdash$$
$$\forall \theta, \phi \in {}^*\mathbb{R}.\ \ \theta \approx \phi \ \rightarrow \ {}^*f'(\theta) \approx {}^*f'(\phi).$$

$\lambda Clam$ first performs rule $(\forall_r)$ twice, and $(\rightarrow_r)$ to yield the conclusion

$${}^*f'(\theta) \approx {}^*f'(\phi).$$

It then makes a case split on $\theta = \phi$. In this case, the conclusion reduces to an identity. In the other case, where $\theta \neq \phi$, we have $\theta - \phi \neq 0$, and then since $\theta \approx \phi$ $\lambda Clam$ evaluates the derivative of ${}^*f$ at θ and ϕ. It also uses rule (5.12) to simplify the expressions. This reasoning yields

$${}^*f'(\theta) \approx \frac{f(\theta) - f(\phi)}{\theta - \phi}$$

and

$$^{*}f'(\phi) \approx \frac{f(\theta) - f(\phi)}{\theta - \phi}$$

hence by transitivity of $\approx$, $^{*}f'(\theta) \approx {}^{*}f'(\phi)$ as required. We refer to the reasoning patterns we observe here as **simplification of derivatives**.

Notice that in this proof-plan the form of the conjecture is different to that of Rolle's Theorem and of the Intermediate Value Theorem. In particular, a case-split for θ is needed, and rules $\forall_r$ and $\rightarrow_r$ have to be applied. These are specific to this theorem and not used in any other proof-plan from the development set.

6.4 Common reasoning patterns

In the presentation of the proofs of Rolle's Theorem and the Intermediate Value Theorem, we observe general reasoning patterns and a high degree of modularity. In this section we highlight these patterns of reasoning, which allow us to construct a set of plan-specifications by which to automate proof-plan construction.

6.4.1 Partitioning function

The first and most apparent common concept in the proofs just presented is the partitioning function, which introduces common structure in the proof-plans for the inductive lemmas. The resulting structure is then encapsulated in the plan-specifications. From the proofs of the development set, the following reasoning patterns are evident:

Case-split analysis As the partitioning function splits intervals in two and chooses a left or a right half according to a partitioning criterion, we must implement a system that can plan proofs involving case analysis. We introduce the notion of a case-split set, which is a set of conditional rewrite rules whose conditions can be shown to be *case complete*. This means that the disjunction of all of the conditions to the rewrite rule are provable.

Division by 2 We notice that in many cases, since the term $\frac{X+Y}{2}$ occurs in the proofs, we use lemmas involving division by two. We incorporate this by adding interactively the set of lemmas shown in section B.3.

Rippling As we are performing inductive proofs, we can use the rippling machinery which already exists in $\lambda Clam$. This allows us to speculate lemmas by analysing the terms in

wave holes. Notice also that we can apply coloured rippling to some inductive proofs as they are mutually recursive and have more than one hypothesis.

Common inductive lemmas

As we can see from the inductive proofs about the intermediate value theorem and Rolle's Theorem, many common lemmas must be proved. The inductive lemmas which are common to both theorems are proved independently of the partitioning criterion, and so could theoretically be proved as theorems of a higher order, more general form of the partitioning function. We have not included such proofs here as they proved too difficult to formulate for an arbitrary number of arguments in $\lambda Clam$, but we discuss some preliminary experimentation in section 6.9.1. Our solution is to introduce plan-specifications which encapsulate the common lemmas, and reasoning patterns which are true of all partitioning functions. This can be seen from the plan-specification represented in figure 6.5.

Mutual induction and coloured rippling

As can be seen from the inductive proofs which establish the outer bounds of the partitions, $\lambda Clam$ has to prove a conjunction of goals. Applying induction to a conjunctions of goals allows $\lambda Clam$ to use the induction hypothesis from each conjunct in the conclusion. Since there is more than one induction hypothesis, coloured rippling can be used. This sort of reasoning corresponds to mutual induction and happens with the left and right points of the partition.

6.4.2 The final stages of the proof-plans

The reasoning steps at the end of the proof-plans contain many reasoning patterns which can be encapsulated. The general idea incorporated in this stage of the proof-plan is to transfer back to the real numbers from the non-standard domain. As the proofs are in a forward direction, we introduce methods for rewriting hypotheses. We notice the following patterns of reasoning:

Witness introduction We introduce a real variable in the hypotheses, infinitely close to the abbreviated variables, and determine the appropriate bounds using the lemmas introduced in section 6.6.2.

Discharging conditions In the hypotheses, there exist definitions which take on the form $A \rightarrow B$ (e.g. uniform continuity), and if we can show A then we can generate the extra hypothesis B. We do this until no more extra hypotheses can be generated.

Simplification of derivatives For Rolle's theorem we obtain an expression for the derivative using the definition for uniform differentiability (6.8). We do this in conjunction with rule (5.12).

Order constraints As can be seen from the proofs, we often have to reason about order constraints on variables to ascertain their sign, or find an infinitesimal neighbourhood in which they exist. We introduce the set of lemmas shown in section B.3 to help us reason about goals involving order relations.

6.5 Obtaining a plan-specification

We present the plan-specifications we developed for the automatic construction of proof-plans for theorems belonging to the same family as those in the development set (i.e. Rolle's Theorem and the Intermediate Value Theorem). In what follows, the graphical description of plan-specifications follows the same annotation scheme as that described in section 5.3 of chapter 5.

6.5.1 Overall plan-specification

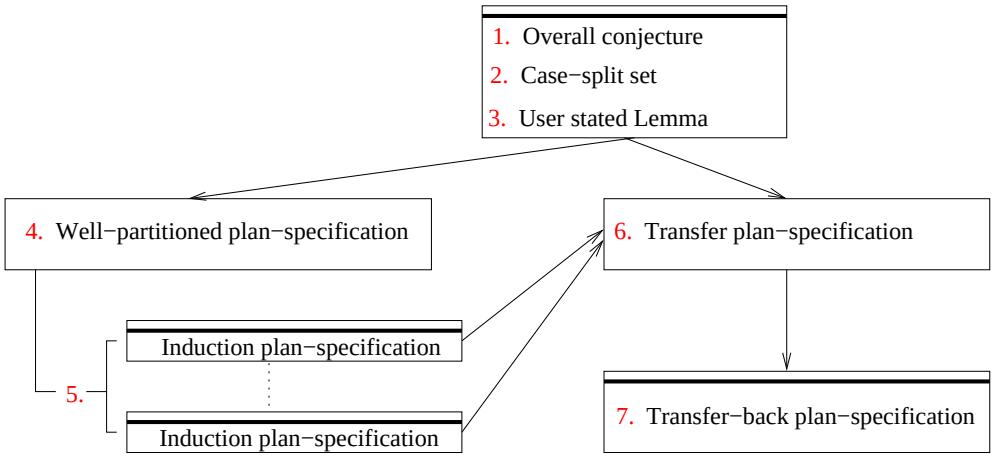

Figure 6.12: Overall plan-specification

The overall plan-specification which accounts for Rolle's Theorem and the Intermediate Value Theorem is described by figure 6.12. The steps that are specified by the overall plan-specification are:

1. Overall conjecture

This is an input to the plan-specification, and refers to the statement of the theorem from analysis, such as Rolle's Theorem.

2. Case-split set

This is an input to the plan-specification, and refers to the name of the rewrite rules representing the partitioning function which is specific to the overall conjecture.

3. User stated lemma

This refers to the inductive lemma which is specific to the partitioning function for the overall conjecture, such as (6.13) for Rolle's Theorem.

4. Well-partitioned plan-specification

Collects the input information and sets up inductive lemmas which are common to all partitioning functions.

5. Induction plan-specification

Produces proof-plans for each of the inductive lemmas produced by the well-partitioned plan-specification.

6. Transfer plan-specification

Abbreviates inductive lemmas, and adds transferred form to hypotheses of the overall conjecture.

7. Transfer-back plan-specification

Produces a proof-plan for the overall conjecture augmented with the new hypotheses.

6.5.2 Induction plan-specification

The induction plan-specification that we use for the inductive lemmas is very similar to that which already exists in $\lambda Clam$ (see figure 3.1 of section 3.2). Figure 6.13 shows our specification for automating the plan construction for the inductive proofs. It may be interesting to compare this with the standard induction plan-specification shown in figure 3.1: ours has the same overall form, as expected, but is more elaborate, producing proof-plans for the inductive lemmas that we are investigating. In the base case we add a case-split after symbolic evaluation which accounts for such base cases as that for Rolle's Theorem, for which we must make a case split on $f'(a) < 0 \vee f'(a) \geq 0$ for example. We embed the hypotheses using coloured rippling

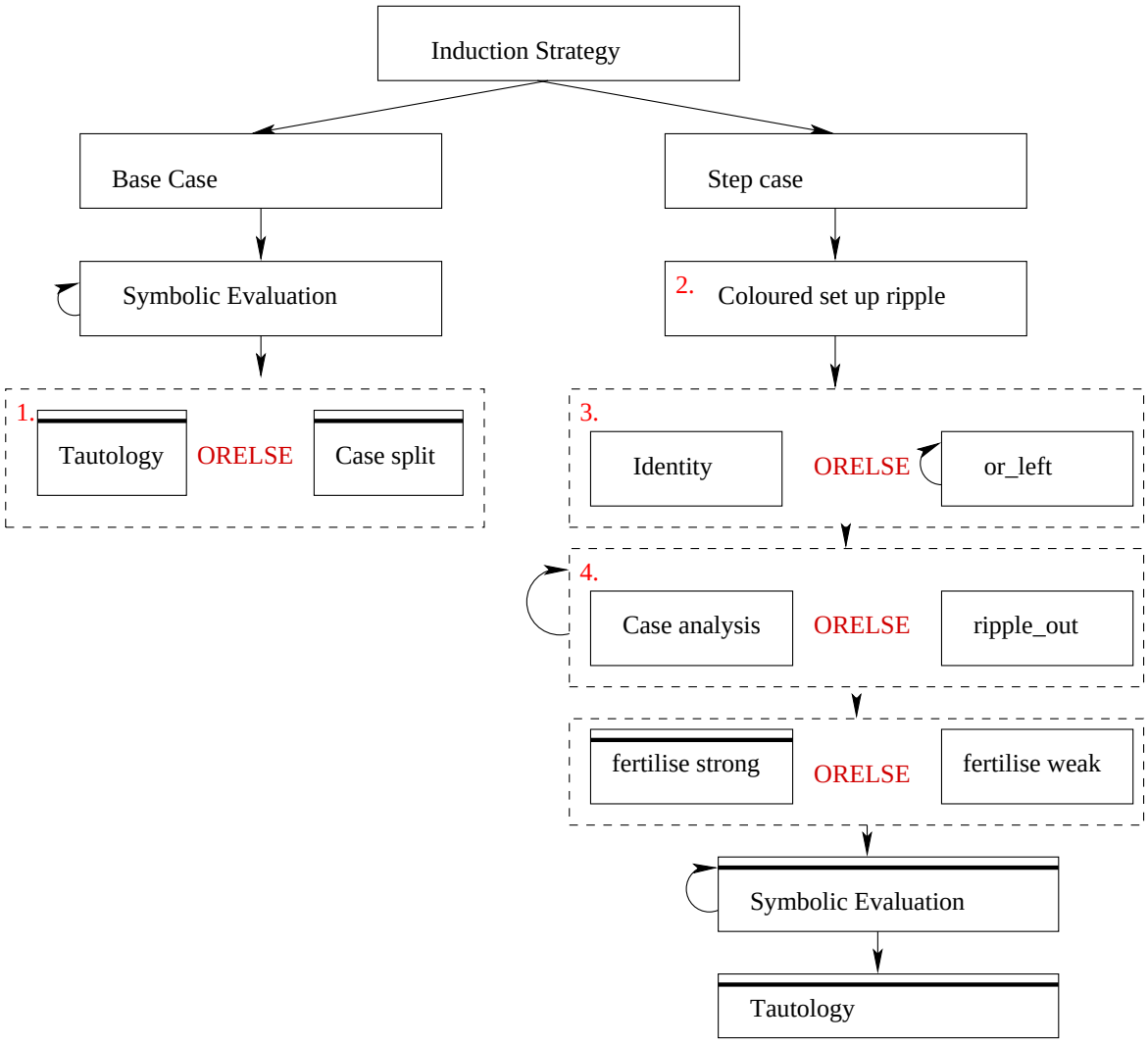

Figure 6.13: Induction plan-specification for proofs of partitioning function

since there may be more than one induction hypothesis. We employ a special case-split method which sets up a conjunction of goals according to the cases of the case-split set chosen. We also add in a part of the plan-specification which removes all disjuncts from the hypotheses if it is not possible to complete a plan leaving such disjuncts in place. Loops in figure 6.13 correspond to the application of a `repeat_meth` methodical. This means that the method is applied repeatedly until it fails, and then the proof-plan proceeds from the next point in the plan-specification.

The important steps of the induction plan-specification are as follows:

1. Tautology orelse Case-split

If after symbolic evaluation, the base case is not provable by the tautology method, then we attempt a case split on the sign of variables in the conclusion. For example if $f'(a) < 0$ in the conclusion, then we make a case-split on $f'(a) \geq 0 \vee f'(a) < 0$.

2. Coloured set up ripple

$\lambda Clam$ embeds all of the induction hypotheses if the goal itself is a conjunction. When rippling takes place each induction hypothesis is attributed with an embedding in the hypothesis.

3. Identity orelse or_left

The identity method leaves the goal unchanged. If the coloured rippling or fertilisation process later fails, then the proof-plan backtracks to this point and applies the $l\vee$ method repeatedly (shown in figure 4.1 of section 4.2.1), and then rippling is re-attempted. This is done since some of the inductive lemmas can be proved by embedding the hypotheses without having to split up the disjuncts in the hypotheses. In the case where this is not possible, the $l\vee$ produces a conjunction of goals on which to apply coloured rippling.

4. Case analysis orelse ripple_out

If $\lambda Clam$ finds a case-split set that applies then a conjunction of goals is set up according to the cases of the case-split and the conclusion rippled out. If a case-split does not apply then $\lambda Clam$ looks for another applicable wave rule to apply.

6.5.3 Transfer-back plan-specification

The final part of the proofs in the development set involves using non-standard analysis to reason about the original analysis theorem (e.g. Rolle's Theorem) using the extra hypotheses

generated by the well-partitioned and transfer plan-specifications (see sections 6.5.4 and 6.5.5 repectively), and to transfer-back to the real numbers. The plan-specification used to direct this part of the proof in $\lambda Clam$ is shown in figure 6.14. Each of the steps employs one of the methods described later in section 6.6.3. We describe the steps as follows:

1. Discharge Conditions

The discharge conditions method generates conditions such as ${}^*f({}^*l(n)) \approx {}^*f({}^*r(n))$ using uniform continuity and the hypothesis ${}^*l(n) \approx {}^*r(n)$.

2. Introduce witness

Method 10 adds a real variable to the hypotheses.

3. Establish bounds

Method 11 establishes bounds on the real variable introduced in step 2.

4. Introduce cases

Method 12 applies rule $l\vee$ to the goal, introducing cases to the proof which correspond to those introduced by the user-defined inductive lemma, for example (6.13).

5. Simplify derivatives

Method 13 instantiates the definition for uniform continuity, simplifies the resulting formula, and adds it to the hypotheses.

6. Order constraints

The order constraints method determines the sign of terms introduced by step 5, introducing new hyperreals and reasoning using transitivity and field equations. If this fails, it uses the lemmas introduced in section 6.6.2 to determine the sign of such terms.

7. Transfer back

Method 14 applies rule (4.40) to determine that the witness introduced in step 1 satisfies the conclusion of the theorem.

6.5.4 Well-partitioned plan-specification

We develop a plan-specification which takes as input a lemma, and a case-split set for which a set of common lemmas can be stated. The plan-specification then applies the induction plan to each of the lemmas. Thus for the Intermediate Value Theorem, for example, we give the well-partitioned plan-specification the names of the rewrite rules and the lemma which are

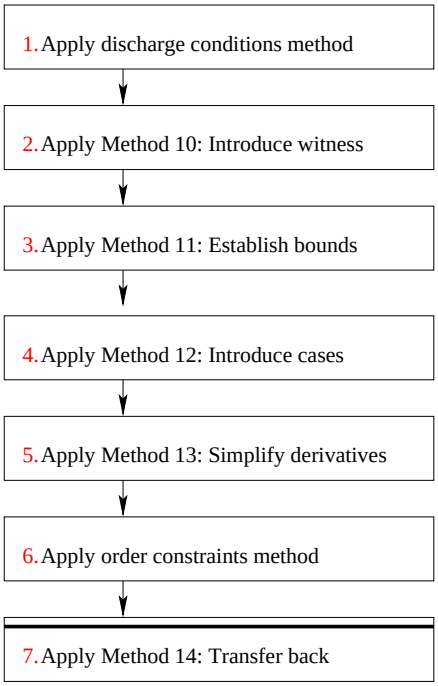

Figure 6.14: The transfer-back plan-specification

specific to the Intermediate Value Theorem. This plan-specification generates the inductive lemmas which are common to any partition function. It generates these lemmas from the input information, namely the names and arguments to two case-split sets, e.g. $\texttt{ivtrec}_\texttt{l}\ f\ a\ b\ c\ n$ and $\texttt{ivtrec}_\texttt{r}\ f\ a\ b\ c\ n$, and the definition of a user-supplied lemma, such as for example

$$\forall n \in \mathbb{N}.\ f(\texttt{ivtrec}_\texttt{r}\ f\ a\ b\ c\ n) \geq \widehat{c} \wedge \widehat{c} \geq f(\texttt{ivtrec}_\texttt{l}\ f\ a\ b\ c\ n)$$

which is the crucial inductive lemma for the Intermediate Value Theorem. The plan-specification now generates a set of inductive lemmas which are common to all partitioning functions, given the names of the rewrite rules provided– in this case $\texttt{ivtrec}_\texttt{l}$ and $\texttt{ivtrec}_\texttt{r}$:

$$\forall n \in \mathbb{N}.\ \texttt{ivtrec}_\texttt{r}\ f\ a\ b\ c\ n - \texttt{ivtrec}_\texttt{l}\ f\ a\ b\ c\ n = \tfrac{b-a}{2^n}$$
$$\forall n \in \mathbb{N}.\ \texttt{ivtrec}_\texttt{r}\ f\ a\ b\ c\ n \geq a \wedge \forall n \in \mathbb{N}.\ \texttt{ivtrec}_\texttt{r}\ f\ a\ b\ c\ n \leq b$$
$$\forall n \in \mathbb{N}.\ \texttt{ivtrec}_\texttt{l}\ f\ a\ b\ c\ n \geq a \wedge \forall n \in \mathbb{N}.\ \texttt{ivtrec}_\texttt{l}\ f\ a\ b\ c\ n \leq b$$
$$\forall n \in \mathbb{N}.\ \texttt{ivtrec}_\texttt{r}\ f\ a\ b\ c\ n > \texttt{ivtrec}_\texttt{l}\ f\ a\ b\ c\ n.$$

6.5.5 Transfer plan-specification

The well-partitioned plan-specification, described in the previous section, generates a set of inductive lemmas which the transfer plan-specification abbreviates as described in section 6.2.2. For example, one of the inductive lemmas specified by the well-partitioned plan-specification for the Intermediate Value Theorem is

$$\forall n \in \mathbb{N}.\ \texttt{ivtrec}_\texttt{r}\ f\ a\ b\ c\ n\ \geq a \tag{6.23}$$

We introduce functions $r : \mathbb{N} \to \mathbb{R}$ and $l : \mathbb{N} \to \mathbb{R}$, where $r(n) = \texttt{ivtrec}_\texttt{r}\ f\ a\ b\ c\ n$ and $l(n) = \texttt{ivtrec}_\texttt{l}\ f\ a\ b\ c\ n$. The inductive lemma (6.23) is then abbreviated to

$$\forall n \in \mathbb{N}.\ \ r(n) \geq a.$$

Next $\lambda Clam$ transfers all of the variables to the non-standard domain, extending the functions r and l and embedding the unquantified real variables. The following is an instance of rule (4.57) from our axiomatisation:

$$\frac{X : \mathbb{R},\ \mathcal{F} : \mathbb{N} \to \mathbb{R} \vdash \forall N \in \mathbb{N}.\ \mathcal{F}(N) \geq X}{X : \mathbb{R},\ \mathcal{F} : \mathbb{N} \to \mathbb{R} \vdash \forall N \in {}^*\mathbb{N}.\ {}^*\mathcal{F}(N) \geq \widehat{X}.}$$

$\lambda Clam$ uses this rule to state the transferred abbreviation of the inductive lemma:

$$\forall n \in {}^*\mathbb{N}.\ \ {}^*r(n) \geq \widehat{a}.$$

6.5.6 A note on the degree of automation

It is not possible to enter Rolle's Theorem as stated in figure (6.5) and yield a proof-plan immediately. There are two specific points in the plan-specification which must be dealt with interactively.

1. The rewrite rules for the partitioning function must be added by hand
2. Any inductive lemmas specific to the partitioning function for the conjecture in question must be added by hand

Once the case split set for the partitioning function has been added to the theory, and the conjecture entered, a crucial inductive lemma must be provided interactively to the system. Once this has been done, the overall plan-specification given in figure 6.12 can produce a complete proof-plan for the Intermediate Value Theorem and for Rolle's Theorem.

6.6 Methods, Critics and Lemmas

We describe the method and critics developed to allow the plan-specifications presented in the previous section to produce proof-plans.

6.6.1 The partitioning function

We describe first the methods and critics developed in order to yield proof-plans for the inductive lemmas about the partitioning function.

Case-split analysis

We introduce a new definition for rewrite rules in $\lambda Clam$ which allows case split analysis to be performed in a principled way. When a case-split set is applied, it is first shown to be complete, meaning that the disjunction of all the cases is provable. Then a conjunction of goals is established, corresponding to each of the cases of the case-split.

Lemma speculation critic

We implement a simple critic which analyses the failure of rippling in an inductive proof. This follows the lemma calculation critic of [Ireland, 1992], and uses the evaluation plan-specification shown in figure 5.5 (of chapter 5) to prove it. For example consider the simple inductive goal for the intermediate value theorem

$$\underline{\texttt{ivtrec}_\texttt{l}\ f\ a\ b\ c\ n + \boxed{\texttt{ivtrec}_\texttt{r}\ f\ a\ b\ c\ n}}_{2}{}^{\uparrow} - \texttt{ivtrec}_\texttt{l}\ f\ a\ b\ c\ n = \boxed{\tfrac{1}{2} \times \underline{b-a}_{2^n}}^{\uparrow}$$

after weak fertilisation and generalisation of common subterms this corresponds to speculating the rewrite rule

$$\underline{Y + \boxed{X}}_{2}{}^{\uparrow} - Y \Rightarrow \boxed{\tfrac{1}{2} \times \boxed{X - Y}}^{\uparrow}.$$

Mutual induction

We implement mutual induction in $\lambda Clam$ by applying induction to a conjunctive goal. This allows us to have more than one induction hypothesis, and we can then apply the built-in embedding method which embeds more than one hypothesis in the conclusion, and then ripples in the normal way with multiple embeddings.

6.6.2 Adding Intermediate lemmas

We add a number of lemmas to the system interactively which are often used in the proof-plans for the inductive lemmas. These are described in full in section B.3.

6.6.3 The transfer-back plan-specification

We introduce the methods which make up the transfer-back plan-specification shown in figure 6.14.

Discharge Conditions

The discharge conditions method assumes we have hyperreals ${}^*l(n)$ and ${}^*r(n)$ introduced by the transfer plan-specification. It looks for definitions of continuity and differentiability in the hypotheses and derives new hypotheses such as ${}^*f({}^*l(n)) \approx f({}^*r(n))$, as for Rolle's Theorem.

Introduce witness

Method 10 uses rule (4.38) to introduce a fresh real variable to the hypotheses. If finds a hyperreal θ in the hypotheses and determines if it is finite by finding bounds, e.g. $\widehat{a} \leq \theta \leq \widehat{b}$ in the case of Rolle's Theorem.

Method 10 Introducing a real witness to the hypotheses

Input: Goal: $\mathtt{H} \vdash G$

Conditions:

hyperreal $\theta \in \mathtt{H}$

$finite(\theta)$

Output: Goal: $\mathtt{H} \cup \{x : \mathbb{R}\} \cup \{\widehat{x} \approx \theta\} \vdash G$

Establish bounds

Method 11 establishes bounds on the introduced real variable, using the lemmas introduced in section 6.6.2.

Introduce cases

Method 12 looks for a hypothesis of the form $D_1 \vee \ldots \vee D_n$ in the hypotheses, and constructs a conjunction of subgoals using rule $l\vee$.

Method 11 Establishing bounds on the real witness

Input: Goal: $\mathtt{H} \vdash G$

Conditions:

$\theta \approx \widehat{x} \in \mathtt{H}$

$\widehat{a} \leq \theta \leq \widehat{b}$

Output: Goal: $\mathtt{H} \cup \{a \leq x \leq b\} \vdash G$

Method 12 Constructing a conjunction of subgoals

Input: Goal: $\mathtt{H} \vdash G$

Conditions:

$D = D_1 \vee \ldots \vee D_n \in \mathtt{H}$

Apply rule $l\vee$ exhaustively

Output: Goal: $\mathtt{H} \cup D_1 \vdash G \quad \ldots \quad \mathtt{H} \cup D_n \vdash G$

Simplify derivatives

Method 13 finds hyperreals θ and ϕ in the hypotheses such that $\theta \approx \phi$ and $\theta \neq \phi$. It then looks for a definition of uniform differentiability for a function f in the hypotheses, determines an expression for $f'(\theta)$, and adds it to the hypotheses.

Method 13 Simplifying derivative expressions

Input: Goal: $\mathtt{H} \vdash G$

Conditions:

hyperreals $\theta, \phi \in \mathtt{H}$

$\theta \approx \phi$

$\theta \neq \phi$

Definition for uniform differentiability for a function f in $\mathtt{H}$

Output: Goal: $\mathtt{H}' \cup \{f'(\theta) \approx \frac{f(\theta)-f(\phi)}{\theta-\phi}\} \vdash G$

Order constraints

The order constraints method uses arithmetic rules (e.g. (B.15)–(B.20) in Appendix B) to rewrite the hypotheses until a term $\mathcal{F}(\theta) \approx \widehat{D}$ can be found. These rules describe the deductions that can be made about terms constrained by ordering relations such as $>$. It uses all of the constraints of the hyperreal variables it can find, performing case splits if the constraint involves

$\geq$ or $\leq$. An outline of the order constraints method is given in sectionintlemap.

As an example of its actions, consider Rolle's Theorem. The term ${}^*l(n) \approx {}^*r(n)$ exists in the hypotheses, and bounds for $f'({}^*l(n))$ and $f'({}^*r(n))$ exist in the cases which are introduced to the hypotheses via Method 12. In one case $f'({}^*l(n)) \geq 0$ and $f'({}^*r(n)) < 0$. The order constraints method performs a case-split for $f'({}^*l(n)) = 0$ or $f'({}^*l(n)) > 0$. The result $f'({}^*l(n)) \approx 0$ is then obtained by applying arithmetic rules such as

$$A \approx B \wedge A < C \wedge B > C \;\rightarrow\; A \approx C.$$

Transfer back

This method uses rule

$$\widehat{X} \approx \widehat{Y} \rightarrow X = Y$$

and the transitivity of $\approx$ to establish that a witness for the theorem in question, and closes the branch of the proof-plan. For example, in Rolle's Theorem we want to establish the existence of a real variable x such that $f'(x) = 0$.

Method 14 Transfer back

Input: Goal: $\mathtt{H} \vdash \exists y \in \mathbb{R}.\ P_1(y) \wedge \ldots \wedge P_n(y)$

Conditions:

real x, hyperreal $\theta \in \mathtt{H}$

$\widehat{x} \approx \theta$

Uniformly continuous function f in hypotheses $\mathtt{H}$

$f(\theta) \approx \widehat{A} \in \mathtt{H}$

add $f(x) = A$ to hypotheses

Uniformly differentiable function f in hypotheses $\mathtt{H}$

$f'(\theta) \approx \widehat{B}$

add $f'(x) = B$ to hypotheses

All conjuncts occur $P_i(x)$ in hypotheses

Output: Branch closed

6.7 Test Set

We describe here two real analysis theorems which we use to test the plan-specifications we constructed using the development set. The first of these is a non-standard version of Rolle's

Theorem which assumes the existence of a maximum point. The second is a generalised version of Rolle's Theorem called the Mean Value Theorem.

6.7.1 Simplified Rolle's Theorem

$$f', f : \mathbb{R} \to \mathbb{R}$$

$$a, b, c : \mathbb{R}$$

$$a < c < b \tag{6.24}$$

$$\forall x, y \in {}^*\mathbb{R}.\ \widehat{a} \leq x \leq \widehat{b} \wedge \widehat{a} \leq y \leq \widehat{b} \wedge x \approx y \ \to\ {}^*f(x) \approx {}^*f(y) \tag{6.25}$$

$$\forall x \in {}^*\mathbb{R}.\ \forall h \in^* \mathbb{R}.\ a \leq x \leq b \wedge h \approx 0 \wedge h \neq 0 \ \to \frac{{}^*f(x+h) - {}^*f(x)}{h} \approx {}^*f'(x) \tag{6.26}$$

$$a < b$$

$$\forall x \in {}^*\mathbb{R}.\ \widehat{a} \leq x \leq \widehat{b} \to {}^*f(x) \leq \widehat{f(c)} \tag{6.27}$$

$$\vdash\ \exists x \in \mathbb{R}.\ a < x < b \wedge f'(x) = 0$$

In this weakened version of Rolle's Theorem, we have already introduced a point c which we know to be the maximum. This mirrors part of the proof in real analysis, where we use the fact that a function attains its maximum on a compact set [Apostol, 1974] to prove Rolle's theorem. There is a dual which states that a function also attains its minimum on a compact set which we do not study here. The important new hypotheses are (6.24) and (6.27). $\lambda Clam$ does not need to generate any new hypotheses by introducing a partition function in this case as we can deduce the result from the existing hypotheses. It applies the transfer-back plan-specification directly. However, the transfer-back plan-specification fails to apply in this case. The discharge conditions method cannot apply as there are no hyperreals in the hypotheses. In order to allow the methods to apply as intended, we must introduce hyperreals to the hypotheses interactively. We introduce hyperreals θ and ϕ such that $\theta < \widehat{c}$, $\phi > \widehat{c}$ and $\theta \approx \phi$. This allows the transfer-back plan-specification to succeed by analysing the sign of the terms ${}^*f'(\theta)$ and ${}^*f'(\phi)$ using the order constraints method, and then using Method 14 to establish that $f'(c) = 0$ over the reals. Now the transfer-back plan-specification succeeds.

The reason for the failure of the transfer-back plan-specification in this case lies in its assumption that hyperreals have already been introduced by the transfer plan-specification. For this characterisation of Rolle's Theorem, we need to add these hypotheses interactively.

6.7.2 Mean Value Theorem

We introduce a generalised version of Rolle's theorem, known as the Mean Value Theorem. We characterise the theorem in non-standard analysis as shown in figure 6.15.

$$f', f : \mathbb{R} \to \mathbb{R}$$

$$a, b : \mathbb{R}$$

$$\forall x, y \in {}^*\mathbb{R}.\ \widehat{a} \leq x \leq \widehat{b} \wedge \widehat{a} \leq y \leq \widehat{b} \wedge x \approx y \ \to\ {}^*f(x) \approx {}^*f(y) \qquad (6.28)$$

$$\forall x \in {}^*\mathbb{R}.\ \forall h \in {}^*\mathbb{R}.\ a \leq x \leq b \wedge h \approx 0 \wedge h \neq 0 \ \to \frac{{}^*f(x+h) - {}^*f(x)}{h} \approx {}^*f'(x) \qquad (6.29)$$

$$a < b$$

$$\vdash\ \exists x \in \mathbb{R}.\ a \leq x \leq b \wedge f'(x) = \frac{f(b)-f(a)}{b-a}$$

Figure 6.15: Our characterisation of the Mean Value Theorem

In most text books on analysis (e.g. [Apostol, 1974]) the Mean Value Theorem is proved by transforming the function f in the statement of Rolle's Theorem. Recall that the conclusion of Rolle's Theorem is

$$\exists x \in \mathbb{R}.\ a \leq x \leq b \wedge f'(x) = 0$$

If we define a function g as

$$g(x) = f(x) - f(a) - (x-a) \times \left(\frac{f(b)-f(a)}{b-a}\right)$$

we yield the expression

$$f'(x) = g'(x) + \frac{f(b)-f(a)}{b-a}.$$

Now notice that $g(a) = g(b) = 0$. We know that g is uniformly continuous and differentiable on the interval $[0,1]$ since it is a linear composition of the functions f and $\lambda x.x$. Now Rolle's theorem tells us there is some point c where $g'(c) = 0$. From this we can determine that $f'(c) = \frac{f(b)-f(a)}{b-a}$ as required.

We present a different proof-plan for the Mean Value Theorem, not assuming Rolle's Theorem and applying the plan-specifications shown in section 6.5.

For a graphical representation of the Mean Value Theorem, see figure 6.16. In Rolle's theorem, as the function f at the point x in the figure had zero derivative, the tests for the partitioning criterion were simple; they depended on the sign of f' and whether the value at

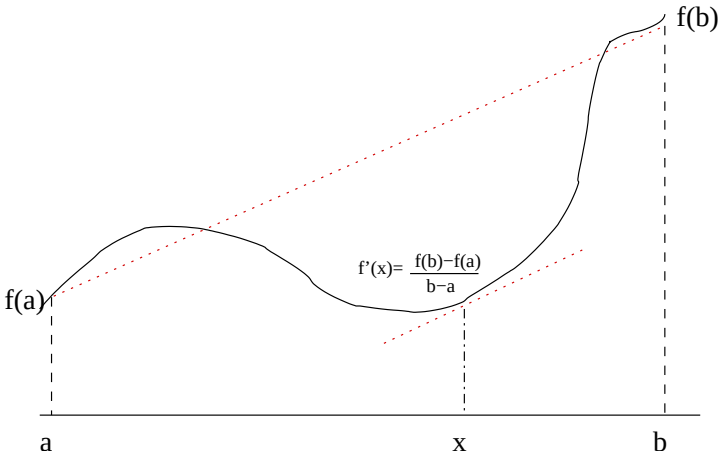

Figure 6.16: The Mean Value Theorem

one end point of the interval was less than or greater than the value at the other end point. In the case of the Mean Value Theorem, this is more general, and the tests for the partitioning criterion become more complicated. More specifically, we define a partitioning criterion, dependent on whether the value of the derivative is greater or less than the value of the derivative at the point *x*- namely $\frac{f(b)-f(a)}{b-a}$. Also we test whether one end point is 'further above' or 'further below' the line produced from the tangent to the point *x* than another. In order to do this we test for end points *r* and *l*

$$f(r) - (\frac{f(b)-f(a)}{b-a} \times r) \geq f(l) - (\frac{f(b)-f(a)}{b-a} \times l)$$

in the case where the right point *r* must be 'further above' the tangent line than the left point *l*.

We omit the presentation of the wave rules here, as they are similar to those of Rolle's Theorem. The common conjectures encapsulated by the well-partitioned plan-specification (see section 6.5.4) can all be planned automatically. We conjecture with pen and paper a corresponding version of the disjunctive lemma introduced for Rolle's Theorem (6.13), and provide it to the overall plan-specification..

The well-partitioned plan-specification and transfer plan-specification all succeed. For simplicity we write

$$r(n) = \texttt{mvtrec}_\texttt{r}\ f\ f'\ a\ b\ n$$
$$l(n) = \texttt{mvtrec}_\texttt{l}\ f\ f'\ a\ b\ n$$
$$X = \tfrac{f(b)-f(a)}{b-a}.$$

The induction plan-specification produces proof-plans automatically for the following lemmas:

$$\forall n \in \mathbb{N}.\ r(n) - l(n) = \frac{b-a}{2^n} \tag{6.30}$$

$$\forall n \in \mathbb{N}.\ r(n) \leq b \wedge r(n) \geq a \tag{6.31}$$

$$\forall n \in \mathbb{N}.\ l(n) \leq b \wedge l(n) \geq a \tag{6.32}$$

$$\forall n \in \mathbb{N}.\ r(n) > l(n) \tag{6.33}$$

$$\begin{aligned}\forall n \in \mathbb{N}.\ & f'(l(n)) \geq X \ \wedge\ f'(r(n)) < X \ \vee \\ & f'(l(n)) < X \ \wedge\ f'(r(n)) \geq X \ \vee \\ & f'(l(n)) \geq X \ \wedge\ f'(r(n)) \geq X \ \wedge\ f(l(n)) - (X \times l(n)) \geq f(r(n)) - (X \times r(n)) \ \vee \\ & f'(l(n)) < X \ \wedge\ f'(r(n)) < X \ \wedge\ f(r(n)) - (X \times r(n)) \geq f(l(n)) - (X \times l(n))\end{aligned} \tag{6.34}$$

The transfer-back plan-specification does not succeed for the Mean Value Theorem as it requires the following two lemmas which are not available to the system:

$$C - E \neq 0 \wedge A - (B \times C) = D - (B \times E) \ \rightarrow\ \tfrac{A-D}{C-E} = B$$

$$C - E \neq 0 \wedge A - (B \times C) < D - (B \times E) \ \rightarrow\ \tfrac{A-D}{C-E} < B.$$

Having added these rules to the system interactively, the plan-specification successfully yields a complete proof-plan.

6.7.3 Simple higher order test

As a simple test of the higher order capabilities of $\lambda Clam$, we introduced a partitioning function with an arbitrary partitioning criterion, which bisects the interval. We introduce a partitioning function which bisects an interval for an arbitrary partitioning criterion $\mathcal{P}$. The partitioning criterion we include in this case is the simplest we can devise – $\mathcal{P} \vee \neg\mathcal{P}$ – i.e. a simple case split such as that for the Intermediate Value Theorem. The wave rules which are introduced, as shown in figure 6.17, allow the well-partitioned plan-specification to yield proof-plans for all of the common inductive lemmas it specifies. This simple example causes no problem because the completeness of the case-split set can be shown purely by assuming the law of the excluded middle.

The induction plan-specification automatically produces proof-plans for the following lemmas:

$$\forall n \in \mathbb{N}.\ \mathtt{horec_r}\ f\ f'\ a\ b\ n\ -\ \mathtt{horec_l}\ f\ f'\ a\ b\ n = \tfrac{b-a}{2^n} \tag{6.35}$$

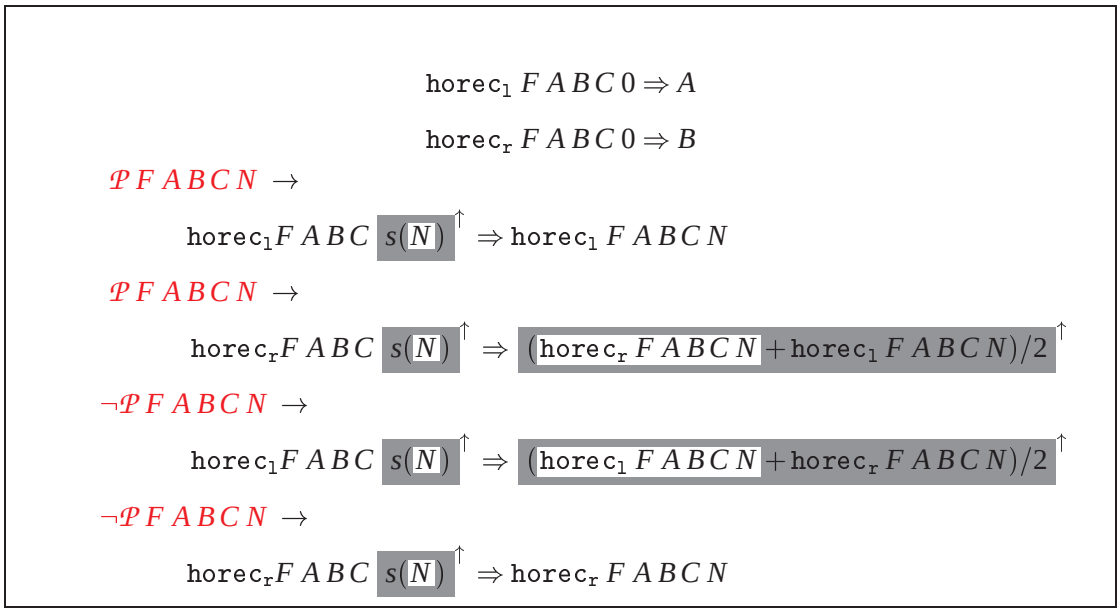

Figure 6.17: The wave rules representing a partitioning function with a simple higher order partitioning criterion

$$\forall n \in \mathbb{N}.\ \mathtt{horec}_r\ f\ f'\ a\ b\ s(n) \leq b \wedge \mathtt{horec}_r\ f\ f'\ a\ b\ n \geq a \tag{6.36}$$

$$\forall n \in \mathbb{N}.\ \mathtt{horec}_1\ f\ f'\ a\ b\ s(n) \leq b \wedge \mathtt{horec}_1\ f\ f'\ a\ b\ n \geq a \tag{6.37}$$

$$\forall n \in \mathbb{N}.\ \mathtt{horec}_r\ f\ f'\ a\ b\ n > \mathtt{horec}_1\ f\ f'\ a\ b\ n \tag{6.38}$$

6.7.4 The trisection method

As another test of the plan-specifications developed, the trisection method for a restricted version of the Intermediate Value Theorem is attempted in $\lambda Clam$.

As described in [Bishop and Bridges, 1985], this method is algorithmically realisable. This means that it is possible for a machine to verify which of the cases of the partitioning criterion apply. For this reason it is an interesting test case, as there is true algorithmic content in the partitioning function (see figure 6.19), unlike with the bisection method.

We restrict the class of functions under consideration to be those which are monotonically increasing, and uniformly continuous on the interval $[a,b]$. We note here that we deal with a function which is monotonically increasing everywhere, when the function should in fact only be monotonically increasing in the interval $[a,b]$; this restriction could easily be lifted.

This restricted characterisation of the Intermediate Value Theorem can be seen in figure

6.18.

$$f : \mathbb{R} \to \mathbb{R}$$
$$a, b, c : \mathbb{R}$$
$$\forall x, y \in \mathbb{R}.\ x < y \to f(x) < f(y) \tag{6.39}$$
$$\forall x, y \in {}^*\mathbb{R}.\ \widehat{a} \leq x \leq \widehat{b} \wedge \widehat{a} \leq y \leq \widehat{b} \wedge x \approx y \ \to\ {}^*f(x) \approx {}^*f(y)$$
$$a \leq b$$
$$f(a) \leq c \leq f(b)$$
$$\vdash \exists x \in \mathbb{R}.\ a \leq x \leq b \wedge f(x) = c$$

Figure 6.18: The characterisation of the Intermediate Value Theorem for use with the trisection method

In order to construct a partitioning criterion for this method, we must rely on the fact that given x, a and b,

$$b > a \ \to\ x > a \vee x < b$$

is constructively true. We thus set up the partitioning function shown in figure 6.19.

```
trirec F A B C 0 = [A,B]
trirec F A B C s(N) =
      let [X,Y]=trirec F A B C N
         in
               if C > F((2X+Y)/3) then [(2X+Y)/3,Y]
          else if C < F((X+2Y)/3) then [X,(X+2Y)/3]
```

Figure 6.19: The partitioning function for the trisection version of the Intermediate Value Theorem

We then attempt to yield proof-plans for inductive lemmas which are almost the same as those for the version of the Intermediate Value Theorem using bisection. We note here the differences in the proof-plans for the inductive lemmas taking particular note of any extra theorems that were needed. We introduce wave rules for $\texttt{trirec}_\texttt{r}$ and $\texttt{trirec}_\texttt{l}$ in the same way as we did for the Intermediate Value Theorem (see figure 6.20).

$$\texttt{trirec}_\texttt{l}\ F\ A\ B\ C\ 0 \Rightarrow A$$

$$\texttt{trirec}_\texttt{r}\ F\ A\ B\ C\ 0 \Rightarrow B$$

$$F(((2 \times \texttt{trirec}_\texttt{l}\ F\ A\ B\ C\ N) + \texttt{trirec}_\texttt{r}\ F\ A\ B\ C\ N)/3) < C \rightarrow$$
$$\texttt{trirec}_\texttt{r} F\ A\ B\ C\ \boxed{s(N)}^{\uparrow} \Rightarrow \texttt{trirec}_\texttt{r}\ F\ A\ B\ C\ N$$

$$F((2 \times (\texttt{trirec}_\texttt{l}\ F\ A\ B\ C\ N) + \texttt{trirec}_\texttt{r}\ F\ A\ B\ C\ N)/3) < C \rightarrow$$
$$\texttt{trirec}_\texttt{l} F\ A\ B\ C\ \boxed{s(N)}^{\uparrow} \Rightarrow \boxed{((2 \times \boxed{\texttt{trirec}_\texttt{l}\ F\ A\ B\ C\ N}) + \texttt{trirec}_\texttt{r}\ F\ A\ B\ C\ N)/3}^{\uparrow}$$

$$F(\texttt{trirec}_\texttt{l}\ F\ A\ B\ C\ N + (2 \times \texttt{trirec}_\texttt{r}\ F\ A\ B\ C\ N)/3) > C \rightarrow$$
$$\texttt{trirec}_\texttt{r} F\ A\ B\ C\ \boxed{s(N)}^{\uparrow} \Rightarrow \boxed{(\texttt{trirec}_\texttt{l}\ F\ A\ B\ C\ N + (2 \times \boxed{\texttt{trirec}_\texttt{r}\ F\ A\ B\ C\ N})/3)}^{\uparrow}$$

$$F(\texttt{trirec}_\texttt{l}\ F\ A\ B\ C\ N + (2 \times \texttt{trirec}_\texttt{r}\ F\ A\ B\ C\ N)/3) > C \rightarrow$$
$$\texttt{trirec}_\texttt{l} F\ A\ B\ C\ \boxed{s(N)}^{\uparrow} \Rightarrow \texttt{trirec}_\texttt{l}\ F\ A\ B\ C\ N$$

Figure 6.20: The wave rules representing the partitioning function for the trisection method for the Intermediate Value Theorem

When we apply the overall plan-specification to the Intermediate Value Theorem using the trisection case-split (see figure 6.20) we find that the induction plan-specification fails at the point where the case-split set needs to be proven to be complete. From figure 6.20 we see that the conditions are

$$F(((2 \times \mathtt{trirec_l}\ F\ A\ B\ C\ N + \mathtt{trirec_r}\ F\ A\ B\ C\ N)/3) < C$$
$$F(\mathtt{tritrec_l}\ F\ A\ B\ C\ N + (2 \times \mathtt{trirec_r}\ F\ A\ B\ C\ N)/3) > C$$

which can only be proved to be complete by using hypothesis (6.39):

$$\forall x, y \in \mathbb{R}.\ \ x < y \rightarrow f(x) < f(y),$$

and the lemma

$$\forall n \in \mathbb{N}.\ \mathtt{trirec_r}\ f\ a\ b\ c\ n\ > \mathtt{trirec_l}\ f\ a\ b\ c\ n \quad . \tag{6.40}$$

This must then be used as an assumption in all subsequent inductive lemmas in order for the case-split set to be used.

In order to solve this problem we construct first a proof-plan for lemma (6.40). During the step case of the proof-plan, we interactively generate the subgoal

$$f : \mathbb{R} \rightarrow \mathbb{R}$$
$$a, b, c : \mathbb{R}$$
$$n : \mathbb{N}$$
$$\mathtt{trirec_r}\ f\ a\ b\ c\ n\ > \mathtt{trirec_l}\ f\ a\ b\ c\ n\ \vdash$$
$$f(\tfrac{(2 \times \mathtt{trirec_l}\ f\ a\ b\ c\ n) + \mathtt{trirec_r}\ f\ a\ b\ c\ n}{3}) < C \vee f(\tfrac{\mathtt{trirec_l}\ f\ a\ b\ c\ n + (2 \times \mathtt{trirec_r}\ f\ a\ b\ c\ n)}{3}) > C$$

which makes use of the induction hypothesis of (6.40) and the rewrite rules

$$F(X) < C \vee F(Y) > C \Rightarrow F(Y) > F(X)$$
$$(X + 2Y)/3 > (2X + Y)/3 \Rightarrow Y > X.$$

We use hypothesis (6.39) as a rewrite rule

$$F(Y) > F(X) \Rightarrow Y > X$$

In order to finally ascertain that the case split is valid, we need to know that the right point of any partition is always greater than the left point. Recall that this was not necessary for the

version of the Intermediate Value Theorem using bisection. The only place in the proof-plan where we use the fact that the function is monotonically increasing (given by hypothesis (6.39)) is in showing that the case-split set is complete.

We need a proof-plan for the lemma

$$\forall n \in \mathbb{N}.\ \texttt{trirec}_\texttt{r}\ f\ a\ b\ c\ n\ -\ \texttt{trirec}_\texttt{l}\ f\ a\ b\ c\ n = (b-a) \times \tfrac{2}{3}^n \tag{6.41}$$

which succeeds because the wave critic is able to speculate the following lemmas

$$\forall X, Y \in \mathbb{R}.\ \ \tfrac{X+2Y}{3} - X = \tfrac{2}{3} \times (Y - X)$$
$$\forall X, Y \in \mathbb{R}.\ \ Y - \tfrac{2X+Y}{3} = \tfrac{2}{3} \times (Y - X)$$

which are crucial to yielding a complete proof-plan. The other inductive lemmas which are planned automatically are

$$\forall n \in \mathbb{N}.\ \texttt{trirec}_\texttt{r}\ f\ a\ b\ c\ n\ \geq a \wedge \forall n \in \mathbb{N}.\ \texttt{trirec}_\texttt{r}\ f\ a\ b\ c\ n\ \leq b \tag{6.42}$$
$$\forall n \in \mathbb{N}.\ \texttt{trirec}_\texttt{l}\ f\ a\ b\ c\ n\ \geq a \wedge \forall n \in \mathbb{N}.\ \texttt{trirec}_\texttt{l}\ f\ a\ b\ c\ n\ \leq b \tag{6.43}$$
$$\forall n \in \mathbb{N}.\ f(\texttt{trirec}_\texttt{r}\ f\ a\ b\ c\ n)\ \geq\ c\ \wedge\ c\ \geq\ f(\texttt{trirec}_\texttt{l}\ f\ a\ b\ c\ n). \tag{6.44}$$

We need to introduce more lemmas interactively in order to yield complete proof-plans for these remaining inductive lemmas. The wave rules for these new lemmas are

$$\frac{2X+Y}{3}^{\uparrow} > Z \vee \frac{2X+Y}{3}^{\uparrow} = Z \Rightarrow \boxed{X > Z \vee X = Z \wedge Y > Z \vee Y = Z}^{\uparrow}$$
$$\frac{2X+Y}{3}^{\uparrow} < Z \vee \frac{2X+Y}{3}^{\uparrow} = Z \Rightarrow \boxed{X < Z \vee X = Z \wedge Y < Z \vee Y = Z}^{\uparrow}.$$

They are the trisection counterparts of the bisection ones presented in section 6.6.2.

In order for the transfer plan-specification to add hypotheses which will allow the transfer-back plan-specification to succeed, we need to ascertain that

$$(\widehat{b} - \widehat{a}) \times (\tfrac{2}{3})^n \approx 0$$

when n is infinite. Once this has been done, the transfer-back plan-specification succeeds in exactly the same way as for the intermediate value theorem.

We needed to make modifications to the plan-specifications in order to yield a complete proof-plan for the Intermediate Value Theorem using the trisection method:

- We alter the well-partitioned plan-specification so that the inductive lemmas are all planned at once using mutual induction. This alteration is a general pattern which would not affect the success of the proof-plan construction for the theorems in the development set.
- We alter the transfer plan-specification so that rule (6.45) is used when adding the transferred versions of the inductive lemmas to the hypotheses.
- We add the rules (6.45) and (6.45) to the system interactively in order to yield proof-plans for the inductive lemmas.

6.8 System Performance and results

We discuss here the performance of our system in tackling complex theorems from real analysis. We give a detailed description of the various aspects of the automation of each theorem. We discuss the search space involved, argue that our system reduces the search space, and give an evaluation of the results obtained. Finally we discuss the evaluation, and make some comments on the proof-plans generated.

In order for us to give a meaningful evaluation, let us first remind ourselves of the claims that we make about constructing proof-plans for non-standard analysis.

Plan reuse

We claim that the techniques we use can be easily and quickly applied to new conjectures,

Search space reduction

We claim that our proof-architecture and plan-specifications reduce the search space,

Readability

We claim that the proof-plans we yield are readable.

6.8.1 Successes and Failures

Tables 6.1 and 6.2 shows the results obtained from the development and testing of the plan-specifications and methods that we have developed. These tables demonstrate the degree of automation for each proof-plan. We record whether a proof-plan was yielded, and the size of the proof-plan in the number of atomic methods fired. We also add how many lemmas were speculated, and how many were added by hand. For the main real analysis theorems,

the inductive lemmas are not included in the number of lemmas added by hand. The relevant inductive lemmas to each theorem are presented in groups under the name of the main theorem. Where $\lambda Clam$ used mutual induction, we present the figures for the proof-plans of the lemmas together. The number of critics fired represents the number of times the lemma calculation critic fired, and the number of speculated lemmas corresponds to the number of lemmas which were correctly calculated. The lemmas which were added by hand for the development set are those in section 6.6.2, which are used throughout all of the proofs. Finally we record the time taken to encode and yield proof-plans for the conjectures in hours.

Conjecture	Proof-plan yielded	Size of proof-plan	No. critics fired	No. spec. lems	No. lems by hand	Dev. time /hours
IVT	yes	8	0	0	2	120
(6.3)	yes	18	1	2	1	36
(6.4),(6.5)	yes	44	0	-	2	40
(6.6)	yes	18	1	1	2	24
Rolle	yes	36	0	0	2	80
(6.9)	yes	18	1	2	1	2
(6.10),(6.11)	yes	44	0	-	2	2
(6.12)	yes	18	1	0	2	2
(6.13)	yes	60	0	0	0	24
(6.14)	yes	16	0	0	3	8

Table 6.1: Results for the development set

Once again the test set in this chapter is relatively small. We were not able to test the plan-specifications on other theorems, but some investigations were made into integration as can be seen in chapter 7.

6.8.2 Search Space

The issue of whether our plan-specifications reduce the search space is important in analysing the success of our proof-planning machinery and plan-specifications. We cannot claim that we have successfully found structure in these proofs, if it turns out that applying all of the lemmas in a more simple search would have resulted in proofs.

In the case of induction, the rippling mechanism is more useful for analysis for the lemma

Conjecture	Proof-plan yielded	Size of proof-plan	No. critics fired	No. spec. lems	No. lems by hand	Dev. time /hours
Sim.Rolle	yes	-	-	-	2	8
MVT	yes	8	0	0	8	12
(6.30)	yes	18	1	2	1	2
(6.31),(6.32)	yes	44	0	-	2	2
(6.33)	yes	18	1	0	2	4
(6.34)	yes	64	0	0	1	12
(6.35)	yes	18	1	2	1	1
(6.36),(6.37)	yes	44	0	-	2	1
(6.38)	yes	18	1	1	2	1
Trisection	yes	8	0	0	3	1
(6.41)	yes	16	1	0	3	24
(6.42),(6.43)	yes	18	1	2	1	12
(6.40)	yes	44	0	-	4	12
(6.41)	yes	18	1	1	4	12

Table 6.2: Results for the test set

calculation critic, but does not significantly reduce the search space as there is only one universally quantified variable. The case analysis allows us to complete the proof-plans in a principled way, and is not included in the usual induction method. In these proof-plans the important piece of reasoning is encapsulated in the lemmas we include by hand in section 6.6.2. The actions of the lemma calculation critic can be reproduced with a generalisation step after weak fertilisation, and in order to plan these proofs at all we need the case analysis. What is crucial to the proofs are the lemmas we supply in 6.6.2.

In the case of the transfer-back plan-specification, we construct three tests to determine the degree to which the search space has been reduced. In these tests we use the reformulation of the theorems using the abbreviated information from the inductive lemmas. Without these it is not possible to yield a proof-plan. We test the transfer-back plan-specification using the reformulation of Rolle's Theorem (see figure 6.11) as a conjecture. The numbers relating to the size of the proof-plan and average branching factor are approximate, since this is just a rough measure of the search space.

Our technique

In our technique, there is no backtracking for Rolle's theorem, and a proof-plan is found at depth 8. This is because the first route found is successful, and the methods used by the plan-specification are very strongly linked to this type of proof.

Naïve strategy 1

The first strategy we tested is an iterative deepening planner, equipped only with a rewriting strategy, and access to the axiomatisation presented in chapter 4. Unsurprisingly, there is an enormous combinatorial explosion, and does not find a proof-plan for Rolle's Theorem even after three days of searching. This is because the introduction of new variables and the order relations. In the proof-plan script which was produced during the experiment, the planner reached a depth of 8 and the number of nodes reached was of the order of 100,000, which denotes an average branching factor of roughly 5.

Naïve strategy 2

The second strategy we tested was a plan-specification which tested all of the atomic methods shown for the transfer-back plan-specification in section 6.6.3, using an iterative deepening search strategy. This was more successful yielding a proof-plan for Rolle's Theorem at a depth of 8. The methods are normally applied in a specific order which is determined by the transfer-back plan-specification shown in figure 6.14. In this case, on account of the iterative deepening planner and the naïve waterfall of methods, many unnecessary nodes were visited. There is more than one way in which the atomic methods can apply at any one point. For example, there may be more than one possible way of discharging a condition from a hypothesis, as is the case with Rolle's Theorem. The total size of the proof-plan is roughly 2000 nodes, giving an average branching factor of about 2.4.

Iterative Deepening with current plan-specification

In this case we use an iterative deepening planner with the plan-specification shown in figure 6.14. Now the only choice points which occur are those that exist within the atomic methods as discussed above. In the case of Rolle's Theorem this happens at 2 specific places:

Simplification of derivatives

We can choose either ${}^*l(n)$ or ${}^*r(n)$ to yield a simplified expression for the derivative.

Order Constraints

We can choose either ${}^*l(n)$ or ${}^*r(n)$ to perform the order constraints.

In this case the number of nodes in the search space is 20 as opposed to 8, since the plan duplicates at these choice points. The branching factor here is negligible, due to the fact that there is very little search involved as the plan-specification itself is very prescriptive,

The plan-specifications we have presented in this chapter provide the planner controlled use of non-terminating rules, and allow manipulation of the hypotheses in such a way that a proof-plan can be yielded. The transfer-back plan-specification introduces heuristics which guide the search to find this step in the proof-plan.

6.8.3 Evaluation

We discuss the results shown in tables 6.1 and 6.2, together with the investigations into the search space, in relation to the criteria set out in section 4.4.3. A similar evaluation scheme is given in [Cantu et al., 1996], where proof-planning was used to automate proof in large hardware verification problems.

Plan-specification criteria

- **Generality**

  Recall from section 4.4.3 that generality must be judged according to how well we can reuse our plan-specifications on new conjectures. Looking at the information in tables 6.1 and 6.2, we notice the difference in development time between the conjectures in the development set and those in the test set. We see that the time taken to develop the inductive lemmas for the Mean Value Theorem and the Higher Order Test is significantly less than for the Intermediate Value Theorem, for which the inductive planning machinery was initially implemented. In the case of trisection, more work must be done as new lemmas are needed, and in particular the completeness of case-split set proves harder to show than in the case of bisection.

  The most significant difference between the test set and development set is in the development time taken to yield a proof-plan from the transfer-back plan-specification. In the case of the trisection for the Intermediate Value Theorem we need only add one lemma, and the transfer-back plan-specification produces exactly the same proof-plan as that for the bisection technique. We require some lemmas to complete a proof-plan for the Mean

Value Theorem, but the development time is significantly less than for Rolle's Theorem. Notice that the number of unspeculated lemmas is increased in the test set, indicating that the plan-specifications are too tailored to the development set.

We can say that the plan-specifications we developed for the development set apply to the theorems in the test set with only slight modification in each case, and hence our claim of reusability is to some extent substantiated. It must be noted however that the plan-specifications are not as general as we would hope. One reason for this is the low number of critics fired as opposed to lemmas we provide by hand. Also, in such a complex proof-structure, it is unsurprising that slight variations in the method will cause a large difference in the plan-specification needed to produce a proof-plan. An example of this is the use of trisection, where a slight difference in the way the interval is partitioned causes complications in proving the inductive lemmas.

- **Intuitiveness**

In order to test the intuitiveness of the proof-plans, we must assess to what extent they follow the steps that a human would perform. Comparing the proof-plans we construct with text-book proofs of the theorems presented here from standard analysis, we cannot claim that the proofs are intuitive. However, we claim that the forward proof-steps which we use in the transfer-back plan-specification exploit the simplicity of non-standard analysis and produce readable and intuitive proof-plans in that the steps performed are not dissimilar from those that a human would perform. The size of the proof-plan is very small, but this is also due to the amount of complexity which is encapsulated in the atomic methods.

- **Simplicity**

The inductive plan-specifications that we develop are simple, and can yield relatively large proof-plans. The simplicity of inductive plan-specifications is well documented in for example [Bundy, 1989]. As we encapsulate most of the reasoning in the atomic methods, the plan-specifications are simple. This is because each atomic method encapsulates many rules that would need to be applied at the object level.

Process criteria

- **Prescriptiveness**

  In order to assess the prescriptiveness of the process of finding a proof-plan, we must analyse the search space examined. We can see from the simple experiments in section 6.8.2 that our plan-specifications perform less search than a naïve search. The best measure of this rests in the size of the search space when our atomic methods are applied in an arbitrary waterfall, as opposed to ordered and structured in the way shown in figure 6.14. This demonstrates that although much of the search is encapsulated within the methods themselves, the ordering of these methods reduces the search space, and hence is a good representation of the structure of proof for this type of conjecture.

- **Efficiency**

  The issue of efficiency is not our prime concern in this thesis, however it is worth noting that the search space is relatively small in these examples, so the computationally expensive parts of the proof-plan happen within the atomic methods.

6.8.4 Comparison with other work

[Gamboa, 1999] tackles the automation of similar proofs in ACL2 . In his thesis, he introduces a proof of the intermediate value theorem, assuming that the endpoints of the initial interval, $[a,b]$, satisfy $f(a) > 0 \wedge f(b) < 0$. He introduces a partitioning function called `find-zero` which takes a natural number *n* and partitions the initial interval into *n* pieces and finds which one of these intervals contains a point where the function is 0. When this *n* becomes infinite it can be shown that this interval is infinitesimal, and the *standard* point in this interval is the point at which the value of the function is 0. This approach uses induction as well. The construction of the relevant lemmas which build up to be able to prove the Intermediate Value Theorem is done by hand, using the axiomatisation introduced. He reuses a lot of the lemmas he proves by showing the same result where the endpoints of the initial interval satisfy $f(a) < 0 \wedge f(b) > 0$. The proofs of the lemmas are fully automated, but the proof of the Intermediate Value Theorem itself requires some hints, which correspond to the structure information which we have encoded in our overall plan-specification given in figure 6.12. This work has the advantage of being proved in an object level theorem prover. The disadvantage is that the

techniques employed in proving the Intermediate Value Theorem are not reused, and hence cannot claim to have generality, although it is unclear how well his techniques would have worked on such as examples as we present here.

In [Fleuriot, 2001a], the Intermediate Value Theorem is proved in Isabelle. In this case the bisection method is used, and the proof follows much the same pattern as the proof-plan we obtain from our plan-specification. The final stages of the proof are more lengthy in Isabelle, as there are many object level steps encapsulated in our atomic methods. In his work, Fleuriot points out the advantage that can be gained from automation. Despite the use of non-standard analysis, Fleuriot notes that a fair amount of interaction needs to happen within Isabelle to prove these involved theorems. Many lemmas are needed and variables still need to be instantiated explicitly. Our proof-planning approach seems to have a higher level of automation and in cases requiring interaction, our machinery provides information that can help us to provide the appropriate lemma to $\lambda Clam$, a facility not available in Isabelle.

6.9 Discussion

In this section we discuss some of the issues with planning proofs in the manner described in this chapter. We discuss some theoretical properties, and possible applications, and discuss the difference between the proofs presented here and the classical real analysis proofs.

6.9.1 Higher order theorems

We notice from the simple higher order test we presented in section 6.7.3, that some of the common structure involved in the inductive proof can be described using the higher order aspects of $\lambda Clam$. We could use these features to construct a higher order version of a partitioning function, where we give the arguments as a list. This way we can argue about partitioning functions with any number of arguments.

Our approach instead favours the proof-planning methodology to allow us to construct a plan-specification which produces proof-plans for each of the important properties that are general to all partitioning functions.

6.9.2 Comparison with real analysis proofs

In constructing proof-plans in the manner shown, we are attempting to show that an algorithm for finding a point with a certain property will converge on that point at infinity. It is not

possible to implement this algorithm on the computable reals as checking whether two arbitrary real quantities are greater or less than each other is undecidable. However, performing the proofs in this manner behaves in an unexpected way in comparison to the real analysis proofs. In Rolle's Theorem for example, we require that the range of the existential variable includes the endpoints, which in fact is too weak a claim. In real analysis it must exist within the open interval. This is because it is possible for a point with zero derivative to exist at the end points, but were it to occur, there would have to be another point with zero derivative in the open interval. As a consequence we must weaken Rolle's Theorem further, since we must assume that the derivative is defined at the end points. The problem is that the "algorithm" that we employ for Rolle's Theorem does not necessarily find this point, perhaps finding the end point instead.

The proofs presented in this chapter, as in chapter 5, use non-standard analysis to establish their results. However, non-standard analysis is used to a lesser degree in the theorems presented in this chapter. A standard version of these proofs would appeal to completeness of the reals. Intuitively the partitioning function provides a sequence of reals. Allow the sequence to be infinite, by transferring to the non-standard domain and using infinite hypernaturals provides us a means of "accruing" the infinite sequence to a point infinitely down. This does rely on the non-standard notion of convergence of a sequence, and hence implicitly uses the completeness of the reals. The partitioning parts of these proofs are entirely standard.

6.9.3 Algorithmic Content

The partitioning functions we which use bisection are only algorithms if we assume a decision procedure for ordering real numbers. However, the trisection version of the Intermediate Value Theorem introduces a partitioning criterion which is constructively provable. In this sense there is computational content in the inductive lemmas for which we obtain proof-plans. However, as we use non-standard analysis to establish the final result, the overall proof-plan does not have immediate computational content.

6.9.4 Rolle's Theorem

We claim that the proof-plan we show for the proof of Rolle's Theorem represents a new method for the proof of a weakened version of Rolle's Theorem. Other partitioning techniques, for example [Abian, 1979], have been attempted. Here, Abian takes an interval (initially $[a, b]$ in our example) and finds the points a_i, p_i, m_i, q_i, b_i which split the interval $[a_i, b_i]$ into four equal

intervals. The recursive function he writes then chooses the successive interval $[a_{s(i)}, b_{s(i)}]$ to be the first of the intervals

$$[a_i, m_i] \quad [p_i, q_i] \quad [m_i, b_i]$$

for which the midpoint which is not exceeded by the midpoint of the other intervals. This ingenious approach contructs a constructive proof which is completed using standard analysis. The general technique is very similar to that which we adopt in this chapter. Our method differs in that the partitioning criterion is dependent on the derivative of the function in question. Had we been aware of this method at the time of writing this thesis, it would have been very instructive to attempt it in $\lambda Clam$. We anticipate that it would not have proved difficult to formulate a partitioning function for this method, and yield a complete proof-plan for Rolle's Theorem.

6.10 Summary

We have presented proof-planning machinery and sample proof-plans for another family of analysis theorems. We incorporated induction into our approach, borrowing ideas from computable analysis. We introduced the notion of a partitioning function, and showed how we yield proof-plans for lemmas about partitioning functions for specific examples. We abbreviated our inductive lemmas and transferred the results to the non-standard domain, where we were able to use non-standard analysis techniques to yield proof-plans for the original analysis theorems.

We believe our technique of combining techniques from computable analysis and non-standard analysis to be novel. In particular we present a novel proof of a weakened version of Rolle's Theorem, and showed that is was possible to use our approach to yield a proof-plan for a restricted version of the Intermediate Value theorem using a trisection technique, which allows the proof to be algorithmically realisable.

Chapter 7

Further Work and Conclusions

In this chapter we present some possible further research directions, indicating some work we have already done to indicate their viability. We also draw some conclusions from the results we achieved in this thesis.

7.1 Further Research

We describe five main areas of research which we have not investigated in this thesis, but which follow on naturally from the work done.

7.1.1 Integration

Following on from the definition of the finite sums given in [Fleuriot, 2001a], we can apply rippling and proof-planning techniques to some theorems about integration. We present here a simple version of an integration theorem using this formalisation.

Integral definitions

We present a simplified definition of the Riemann integral over the reals

$$\int_a^b f(x)dx = I \iff \lim_{h\to 0}\Big(\sum_{k=1}^{k=\lfloor\frac{b-a}{h}\rfloor} f(a+kh)\cdot h\Big) = I$$

which using a more simple functional notation can be rewritten to

$$\int_a^b f(x)dx = I \iff \lim_{n\to\infty} sum(1,n,\lambda x.\, f(x),a,b) = I.$$

where $f : \mathbb{R} \to \mathbb{R}$, and $a, b \in \mathbb{N}$ and $I \in \mathbb{R}$. The function *sum* takes two natural numbers, which represent the limits of the sum, a real-valued function, and two real numbers. The function argument is evaluated in the interval given by the real numbered arguments, at the positions given by the natural numbered arguments, and a sum is taken.

The first two arguments represent the limits to the summation, within the specified end-points. The function is We borrow some of the theorems proved in [Fleuriot, 2001a]:

$$\forall N. inft(N) \to hrsum(1, N, {}^*\lambda x. f(x), \widehat{a}, \widehat{b}) \approx \widehat{I} \iff {}^*((\lambda n. sum(1, n, \lambda x. f(x), a, b)))N \approx \widehat{I}. \quad (7.1)$$

Here the type of hrsum is ${}^*\mathbb{N} \to {}^*\mathbb{N} \to ({}^*\mathbb{R} \to {}^*\mathbb{R}) \to {}^*\mathbb{R} \to {}^*\mathbb{R} \to {}^*\mathbb{R}$, and the subtyping predicate $inft(N)$ indicates that N is an infinite hypernatural. This allows us to write the following unfolding rule

$$NSint(a, b, \lambda x. f(x)) = I \iff \forall N.\ inft(N) \to hrsum(1, N, {}^*\lambda x. f(x), \widehat{a}, \widehat{b}) \approx \widehat{I}$$

where the information about how sums work is contained within the *hrsum* function, and *NSint* is a non-standard extension of integral.

7.1.2 Theorem

The following is a theorem over the reals:

$$\int_a^b f(x)dx + \int_a^b g(x)dx = \int_a^b f(x) + g(x)dx$$

and we want to be able to prove it over the hyperreals. So state the theorem as

$$NSint(a, b, \lambda x. f(x)) = I_f \wedge NSint(a, b, \lambda x. g(x)) = I_g \vdash NSint(a, b, \lambda x. (f(x) + g(x))) = I_f + I_g$$

and rewrite as

$$\forall N.\ inft(N) \to hrsum(1, N, {}^*\lambda x. f(x), \widehat{a}, \widehat{b}) \approx \widehat{I_f}$$
$$\forall N.\ inft(N) \to hrsum(1, N, {}^*\lambda x. g(x), \widehat{a}, \widehat{b}) \approx \widehat{I_g}$$
$$\vdash \forall N.\ inft(N) \to hrsum(1, N, {}^*\lambda x. (f(x) + g(x)), \widehat{a}, \widehat{b}) \approx \widehat{I_f + I_g}.$$

Rewriting using rule 7.1 we yield

$$\forall N.\ inft(N) \to (\lambda n. sum(1, n, \lambda x. f(x), a, b)^*)N \approx \widehat{I_f}$$
$$\forall N.\ inft(N) \to (\lambda n. sum(1, n, \lambda x. g(x), a, b)^*)N \approx \widehat{I_g}$$
$$\vdash \forall N.\ inft(N) \to ((\lambda n. sum(1, n, \lambda x. (f(x) + g(x)), a, b))^*)N \approx \widehat{I_f + I_g}.$$

Now it is possible to set up the coloured rippling process. After $\forall$ introduction, choosing an arbitrary m for N, the conclusion becomes

$$inft(m) \rightarrow {}^*(\lambda n.sum(1,n,(\lambda x.(\boxed{f(x)+g(x)}^{\uparrow}),a,b)))m \approx (\boxed{\widehat{I_f+I_g}}^{\uparrow}).$$

Now we use the rules

$$sum(M,N,\lambda x.\boxed{F(x)+G(x)}^{\uparrow},A,B) \Rightarrow \boxed{sum(M,N,\lambda x.F(x),A,B)+sum(M,N,\lambda x.G(x),A,B)}^{\uparrow}$$

$$\boxed{\widehat{I_f+I_g}}^{\uparrow} \Rightarrow \boxed{\widehat{I_f}+\widehat{I_g}}^{\uparrow}$$

to ripple the conclusion to

$$inft(m) \rightarrow {}^*((\lambda n.(\boxed{sum(1,n,\lambda x.f(x),a,b)+sum(1,n,\lambda x.g(x),a,b)}^{\uparrow}))m \approx \boxed{\widehat{I_f}+\widehat{I_g}}^{\uparrow}.$$

Now appeal to the wave rules

$$\lambda x.(\boxed{F(x)+G(x)}^{\uparrow}) \Rightarrow \boxed{\lambda x.F(x)+\lambda x.G(x)}^{\uparrow}$$

$${}^*(\boxed{\lambda x.F(x)+\lambda x.G(x)}^{\uparrow}) \Rightarrow \boxed{{}^*(\lambda x.F(x))+{}^*(\lambda x.G(x))}^{\uparrow}$$

$$(\boxed{{}^*(\lambda x.F(x))+{}^*(\lambda x.G(x))}^{\uparrow})N \Rightarrow \boxed{({}^*(\lambda x.F(x)))N+({}^*(\lambda x.G(x)))N}^{\uparrow}$$

which ripples the conclusion to

$$inft(m) \rightarrow \boxed{{}^*(\lambda n.sum(1,n,\lambda x.f(x),a,b))m+{}^*(\lambda n.sum(1,n,\lambda x.f(x),a,b))m}^{\uparrow} \approx \boxed{\widehat{I_f}+\widehat{I_g}}^{\uparrow}.$$

Now it is possible to use the two familiar wave rules in order to finish the proof

$$\boxed{X+A}^{\uparrow} \approx \boxed{Y+B}^{\uparrow} \Rightarrow \boxed{X \approx Y \wedge A \approx B}^{\uparrow}$$

$$A \rightarrow \boxed{B \wedge C}^{\uparrow} \Rightarrow \boxed{A \rightarrow B \wedge A \rightarrow C}^{\uparrow}$$

which finally yields the conclusion

$$\boxed{inft(m) \rightarrow (\lambda n.sum(1,n,\lambda x.f(x),a,b)^*)m \approx {}^*I_f \wedge inft(m) \rightarrow (\lambda n.sum(1,n,\lambda x.g(x),a,b)^*)m \approx {}^*I_g}^{\uparrow}$$

at which point strong fertilisation applies, as each wave hole is an instance of a hypothesis. We yielded a proof-plan for this theorem in $\lambda Clam$, but did not pursue the research avenue any further. It would be interesting to see to what extent more complex integration theorems such as integration by parts could be tackled by $\lambda Clam$.

7.1.3 Verifying algorithms

As described in [Harrison, 1999], there has been a significant amount of work done in verifying correctness properties of floating point formalisations. As mentioned, the work described in chapter 6 cannot truly be called algorithm verification, as there is no decision procedure for equality over the reals. With the exception of the trisection method, there is no algorithmic content to any of the proofs. As described in [Chippendale, 1995], it is possible to combine proof-planning and computable analysis. A further interesting avenue of research would be to construct such proofs using floating point number systems. It is not clear at this point what role non-standard analysis would play in such proofs.

7.1.4 Object-level proofs

In order to justify our work as helping in the automation of proof, we must be able to execute the proof-plans at the object-level. As we have described the methods we use are complex and do not correspond exactly to the application of one rule at the object-level. We have presented an axiomatisation for our system and have indicated how the methods correspond to applications of these axioms.

We would like to be able to attach the work performed by Fleuriot in Isabelle/HOL to the system we have devised in order to yield object-level proofs [Fleuriot, 2001a]. This is of particular interest since we follow a similar formalisation for non-standard analysis, also using the transfer theorem in both directions in our proof-plans.

As mentioned in section 3.2, methods comprise of a number of "slots". One of these corresponds to the name of a tactic in an object-level theorem prover. This communication has successfully been achieved in the Clam/HOL project [Boulton et al., 1998], where a new tactic `clam_tac` was implemented in HOL which called the Clam proof-planner with inductive conjectures. Using the HOL tactic name in the slot of each method, the proof-plan returned a tree of tactic applications to HOL, which automatically proved the inductive conjecture.

It is feasible that tactics could be written in Isabelle/HOL which correspond to the methods we have implemented. It would then be possible to automate proofs at the object-level by attaching these tactic names to the tactic slot in each atomic method. We have focussed our research on finding structure in proof at the proof-planning level, but a natural extension to our work would be to implement such tactics in Isabelle/HOL.

7.1.5 Mathematical assistant

A further area of research pertaining to this thesis would be to take the techniques presented in [Heneveld et al., 2001] and to apply them to the problem of finding proofs for the continuity of specific functions using non-standard analysis. This work implements methods in $\lambda Clam$ which mirror the general techniques used by students in solving integral and differentiation problems. In particular, a best-first search is employed to choose the best method to apply at each stage according how suitable its application is. The motivation is to use the system as a mathematical assistant, and to model human reasoning in a particular mathematical domain.

The ΩMEGA system [Benzmüller et al., 1997] has many examples of functions which it proves to be continuous using standard analysis, advertising this work in the context of a mathematical assistant. In order to use the approach described in [Heneveld et al., 2001] to the realm of analysis, the techniques would mirror those used in chapter 5, and in particular the use of Method 7, where infinitesimal terms are eliminated from goals. The work of Heneveld is particularly interesting here as we would be able to use $\lambda Clam$ as a basis for learning how humans reason about such proofs, and may even be able to make a comparison between standard and non-standard style proofs.

7.1.6 Non-standard annotation

A final research direction would be to devise some annotation for the elements of non-standard analysis which do not appear in standard analysis. The motivation for this came from chapter 6 where we establish a result in the reals by moving out of the non-standard model in which we have been reasoning. Specifically, when we establish a result such as

$$\widehat{f(x)} \approx \widehat{c}$$

we yield the result

$$f(x) = c$$

using rule 4.40 from the axiomatisation. If we write $\approx$ as an annotated non-standard version of equality, for example $\stackrel{\sim}{=}$, we could introduce some rippling style annotation for each of the non-standard elements, and represent this rule

$$\widehat{f(x)} \stackrel{\sim}{=} \widehat{c} \ .$$

Similar annotations can be made for ${}^*f$ for example. The idea would then be that in non-standard analysis proofs, the skeleton of the term would correspond to some structure in the real

model, and the manipulations we perform in the non-standard model would also manipulate the wave-fronts. The extensional representation of non-standard analysis which we take here would allow us to reason with this annotation, but we have not investigated this far enough to see whether it would be feasible or not.

7.2 Concluding remarks

The work presented here has been analysed in the evaluation sections of chapters 5 and 6. It remains for us to analyse whether we achieved our research hypothesis as described in section 4.1.1, and to describe what the possible repercussions of this work are on both non-standard analysis and proof-planning.

7.2.1 Research Hypothesis

Recall from chapter 4 that we state our research hypothesis as

> Through proof-planning we arrive at intuitive and successful representations of the structure of proof in non-standard analysis.

We believe that this has been achieved in this work. We have yielded complete proof-plans, sometimes with full automation, for some complicated theorems from real analysis. The degree of automation is analysed in sections 5.6.3 and 6.8.3, and we have shown that we have been successful in producing a representation for the structure of proof in non-standard analysis. We have achieved a substantial degree of automation in the proof-plans. The other automated systems which achieve automation of the proofs we study are those presented in [Bledsoe and Ballantyne, 1977] and [Gamboa, 1999].

[Bledsoe and Ballantyne, 1977] achieves full automation of some complicated analysis theorems such as the chain rule. This work uses a resolution style theorem prover together with an axiomatisation of non-standard analysis. Our system is able to automate the construction of proof-plans for the same complicated theorems. However, through our use of critics and domain-specific methods, we produce a much more readable description of the proofs. We also produce proof-plans for a different family of theorems, such as Rolle's Theorem, which Bledsoe's system is not capable of.

[Gamboa, 1999] presents a system which uses Nelson's Internal Set Theory [Nelson, 1977] to define functions using Taylor series and prove properties about them when the number of terms becomes infinite. He presents a proof of the Intermediate Value Theorem which is fully

automated, given the definition of a partitioning function. The version he presents uses a different partitioning function from ours, but the general techniques and level of automation are comparable. The advantage of our development over Gamboa's lies in the encoding of reasoning patterns into plan-specification which are applicable to other, more complicated theorem such as Rolle's Theorem. We have shown that the approach is generalisable to other theorems, requiring only slight modification to the plan-specification or adding some lemmas.

[Fleuriot, 2001a] proves some of our theorems in Isabelle/HOL. In the interactive proofs he presents, he shows how some difficult steps need to be performed with significant user intervention in order to yield proofs. Although non-standard analysis provides a more algebraic formulation of such theorems as the chain rule, difficult proof-steps still need to be performed. For example, instantiations for universally quantified variables in the hypotheses of the sequent still have to be determined. We have implemented critics and methods which help us find these instantiations, and automate difficult proofs. Also we have implemented other useful features, such as lemma speculation. Fleuriot also presents a proof of the Intermediate Value Theorem which has the same approach as ours. We have fully automated the proof-plan construction for this theorem, and have shown that the approach applies easily to other theorems.

The question of intuition is much harder to investigate without a study of human proofs from this domain. What we can claim is that proof-planning enables the construction of methods which correspond to our reasoning patterns, and of critics which correspond to an analysis of failure of these reasoning patterns. We can contrast with the work presented in [Bledsoe and Ballantyne, 1977] for example, where successful proofs were yielded, but were done so in a resolution theorem prover whose proof steps do not correspond to human style reasoning.

7.2.2 Non-standard analysis

Our work has shown that is is possible to reproduce reasoning patterns from non-standard analysis without using any special techniques, such as the limit heuristic and constraint solver that are used in the ΩMEGA system. The more algebraic formulation of the notion of continuity allows the proofs to have a simpler common structure, and in general a very similar way of completing proofs. It certainly can be claimed that the proof patterns of non-standard analysis are less complicated than those of standard analysis, although we cannot claim that one is more *intuitive* than the other since in order to yield the simpler proof structure we must reason in a new number system which does not correspond to the intuition we gain from learning about

the real numbers at school.

7.2.3 Proof-planning

We have achieved a substantial amount of automation for a number of complex analysis theorems. We have exploited the existing proof-planning machinery in λ*Clam* and enhanced the system with our own work, in order to achieve an understanding of the structure of proof for a new domain of mathematics.

We have introduced a system for lemma speculation in λ*Clam* which is not available in other theorem provers or proof-planners. In section 5.4 we implemented a number of critics which are capable of lemma speculation. Our system for lemma speculation is significantly more general than those which have been introduced before. In our system we use the information we gain from coloured rippling to analyse the types of the terms in wave holes, and suggest a lemma which will "join" the wave holes in the conclusion. In the embedding critic we introduce (see section 5.4.1), we introduce a patch which is capable of guessing instantiations to universally quantified variables in the hypotheses and speculating lemmas accordingly, without instantiating the variables in the hypotheses. It must be noted though that this critic, along with some of the methods that we introduce, is domain specific and looks for structures which we expect to find in analysis theorems using definitions from non-standard analysis.

λ*Clam* has been enhanced by the machinery we have introduced. We have introduced case-split sets, coloured rippling and lemma speculation via critics. Also we have provided a set of methods by which we can construct proof-plans for analysis theorems using non-standard definitions in λ*Clam*.

We claim that we have not only incorporated new ideas in λ*Clam*, such as our implementation of lemma-speculation, but that they are useful techniques which could be employed by other theorem proving and proof-planning systems. One such system is IsaPlanner [Dixon and Fleuriot, 2003] which incorporates proof-planning into Isabelle.

7.3 Suggested extensions to the work

We discuss in this section some possible extensions to the work which are too involved to have attempted during the course of research.

7.3.1 Object-level proofs

The major issue with the work discussed is that of soundness. Since we are not producing object-level proofs, it is important that we argue for the soundness of the proofs via our axiomatisation. However, some mistakes in the proofs were discovered upon close inspection due to mistakes in the axiomatisation (see comments in section 5.5.2).

Ideally we would build up an axiomatisation in a formal framework such as Isabelle/HOL. This was not done in the work presented in this thesis as it is a lengthy undertaking in itself to write the tactics in the object-logic which correspond to the methods in the proof-planner. In this section we give an overview of the work required to complete such a task.

The proof-planner performs search on the proof at an abstracted level, and the resulting output proof-plan should execute a proof at the object-level, by associating methods at the proof-planning level with tactics, or sequences of tactics at the object-level. We describe the stages which need to be implemented and performed in order to automate such proofs at the object-level.

Validation of axiomatisation

In order to validate the proofs for our system using an object-level we need to construct an axiomatisation which can be validated from a fundamental formalisation such as that performed in [Fleuriot and Paulson, 2000]. Using this work, the axioms presented in chapter 4 can all be proven in the more fundamental logic of Isabelle/HOL.

Unfolding compound methods

Compound methods comprise of a set of atomic methods ordered using methodicals. In order to produce a tactic at the object-level, the proof-plan which was yielded by applying the plan-specification to the theorem determines how each compound method is unfolded into a sequence of atomic method applications.

Construction of tactics

When atomic methods are used in $\lambda Clam$ we can attach a tactic name in an object-level theorem prover to the method which performs the same operation. Typically for inference rules each method application corresponds exactly to the inference rule at the object-level. However for more complicated atomic methods which perform several rewriting steps or inference rules.

For example consider atomic method 6 of chapter 5 which simplifies the goal of a theorem stated in non-standard analysis. With this method we exhaustively rewrite the goal according to a particular set of rewrite-rules. In order to translate this into a number of application of rewrite-rules at the object-level we must return the precise rules which were used and in which order. This means that we must construct a tactical comprising of the applications of rules used by the method to simplify the goal.

Critics

Critics pose the greatest problem in constructing object-level proofs from proof-plans. Some critics perform plan transformation, and some perform goal transformation but require extra goals to be performed. Although the plan-specifications do not reflect any plan-transormation, the final proof-plans show the complete shape of the proof.

In general the critics we have introduced discover lemmas which are needed in order to yield a complete proof. These lemmas need to be proved and then the cut rule of inference needs to be employed in order to make use of the discovered lemma.

7.3.2 The issue of human intervention

We have developed various techniques in this thesis for automating the construction of proof-plans for the types of theorem we study in this thesis. In particular we have developed some techniques for automating the discovery of lemmas. There are however two issues concerning human intervention that we discuss here.

Complete automation of lemma discovery

Although we have provided many mechanisms for discovering lemmas automatically, there are still several lemmas which had to be introduced interactively, such as those introduced for the chain rule (see equation (5.10), section 5.2.1). In general it is a very difficult problem to automatically generate required lemmas.

Calculating the correct lemma to allow a proof to proceed often requires some expansion of the term structure, which introduces an infinite branch point in the search space. This is because the rule required can introduce new variables which can be instantiated to any term of the correct type. Also, in order to use a rule which expands the term structure, we reduce any limitation on the rewrite rule used, and so many more rewrite rules can potentially apply. In order to speculate a rule which is correct, which allows the proof to succeed, we use heuristics

to reduce to choices in the search space. To obviate the need for heuristics, and to come up with a general scheme for automating such lemma discovery, a much more involved piece of research is required than was possible during the course of this work.

Automation of non-standard formulations

When we enter theorems into the $\lambda Clam$ system, we enter fully expanded non-standard formulations. A valid criticism of this approach is that it would be possible to automate this expansion from a standard formulation of the theorems. As can be seen from section 2.4.2, there are macros for expanding non-standard characterisations of such notions of limit and continuous functions. This would facilitate a more convenient way of entering theorems into $\lambda Clam$. This is a relatively simple operation, and could easily be implemented, but would not have contributed to the main research problems of the thesis that we wanted to address.

7.3.3 The value of the work done

Two major areas of work to which this thesis intended to contribute were the automation of non-standard analysis proofs and to the elucidation of such proofs using proof-planning.

We have contributed to the automation of proofs in non-standard analysis by designing methods and critics which exploit the algebraic nature of non-standard analysis, as discussed in chapters 5 and 6. It must be noted that the lack of object-level proofs reduces the level of contribution. Producing object-level proofs in a fully expansive theorem proved such as Isabelle/HOL would have obviated any doubts about correctness which can always be present using just proof-planning. We have attempted to argue throughout for correctness although as can be seen from section 5.5.2 mistakes can easily be made.

We claim to have enhanced the readability of proofs in non-standard analysis, using the proof-plans that have resulted from this work. We cannot claim to have produced object-level proofs, so a direct comparison is not possible, but we can claim to have produced more readable proof-plans that the proofs created by Bledsoe's work [Bledsoe and Ballantyne, 1977], since the resolution style proofs can be difficult to interpret.

Appendix A

Sample plan-specifications and output

We present some sample plan-specifications as they are written in λ*Clam* , and show some edited output. The output for the theorems shown is very long, so we include only the important parts. Also we introduce some latex symbols to make the output more readable.

We first show the overall plan-specification for the limit problems we present in chapter 5, and then give an example of some output for the chain rule. We also show an extract from a proof-plan for the chain rule drawn by the XBarnacle front end [Lowe and Duncan, 1997] for an earlier version of λ*Clam* . We also show the outermost plan-specification defined for the theorems we introduced in chapter 6, and show the output for Rolle's Theorem.

A.1 Chain Rule

In λ*Clam* , the plan-specifications, as shown in section 5.3, are represented by compound methods. This denotes the theory name, the name of the compound method, a specification of its actions written in methodical language, an address which is left uninstantiated here, and a set of preconditions, which for plan-specifications is set to `true`. The compound method shown in figure A.1 corresponds to the outermost plan-specification described by figure 5.1 in section 5.3. The `patch_meth` methodical corresponds to the explicit attachment of a critic patch plan-specification to a method.

The chain rule is tackled using the outermost compound method. We show here an edited presentation of the output from λ*Clam* , where superfluous information such as embeddings are suppressed and mathematical symbols such as ∀ are used where possible. The presentation of a proof of the chain rule given in section 5.2.1 follows the output from λ*Clam* , which is shown here. λ*Clam* starts with the initial goal.

```
compound nsaconjectures nsa_top_meth_ripple_critics
(repeat_meth
  (then_meth tautology
  (then_meth (patch_meth set_up_ripple embed_critic_strat)
  (then_meth
  (repeat_meth
    (or_else_meth
      (patch_meth (wave_method outward R1) wave_critic_strat)
      (patch_meth (cond_wave_method outward R2) wave_critic_strat)))
  (then_meth (patch_meth fertilise fert_critic_strat))))))
  _
  true.
```

Figure A.1: The outermost compound method for the limit conjectures

```
nsa_plan nsa_top_meth_ripple_critics chainrule.

x:ℝ, d1:ℝ, d2:ℝ, f:ℝ→ℝ, g:ℝ→ℝ
∀ H ∈ *ℝ. (H ≈ 0 ∧ H ≠ 0) →
      H^-1 × (ext f) ((emb (g x))+H)-(emb (f (g x)))  ≈ (emb d1)
∀ H ∈ *ℝ. (H ≈ 0 ∧ H ≠ 0) →
      H^-1 × (ext g) ((emb x)+H)-(emb (g x))  ≈ (emb d2)
⊢
∀ H ∈ *ℝ. (H ≈ 0 ∧ H ≠ 0) →
      H^-1 × (ext f) ((ext g) ((emb x)+H))-(emb (f (g x))) ≈ (emb (d1 × d2))
```

to which the tautology method is first applied. This fails and then attempts the method `set_up_ripple` on the goal, which also fails. So, its preconditions are tested by a meta-interpreter, and the pre-conditions which failed trigger a patching strategy.

```
Attempting...
(patch_meth set_up_ripple embed_critic_strat)

strip_forall_embeds _H _E _S _G     failed

attempting atomic critic
pop_critic _
```

```
pop_critic _Address
succeeded

attempting atomic critic
embedding_failure _ _
embedding_failure _Hyps _Goal
succeeded

attempting atomic critic
open_node
open_node
succeeded
```

At this point the embedding critic plan-specification, as shown in figure 5.2 of section 5.3, applies.

```
attempting atomic critic
emb_spec _ _ _ _ _ _ _
emb_spec _Pos _Subst A × B^-1  (A × H^-1) × (H × B^-1)  H ≠ 0  _Goal _NGoal
succeeded

attempting atomic critic
continue_crit (M (pair_meth eval_meth nsa_top_meth_ripple_critics) M)
continue_crit (M (pair_meth eval_meth nsa_top_meth_ripple_critics) M)
succeeded
```

Now once a proof-plan has been found for the speculated lemma given by the `emb_spec`, embedding can take place and the proof-plan can proceed. Then at the end of the proof-plan, when fertilisation is attempted, since there are mismatching sinks, the fertilisation critic suggests adding some information to the hypotheses. This corresponds to the plan-specification given by figure 5.4 of section 5.3.

```
Attempting...
(patch_meth fertilise fert_critic_strat)

sink_match G H E
failed
.
.
.
```

```
attempting atomic critic
fert_spec _ _ _ _
fert_spec _Hyp _G [(ext g) ((emb x) + H) - (emb (g x)) ≈ 0,
        (ext g) ((emb x) + H) - (emb (g x)) = 0  ∨
                    (ext g) ((emb x) + H) - (emb (g x)) ≠ 0] _NewG
succeeded

attempting atomic critic
roll_back_to_crit _ nil
roll_back_to_crit nsa_top_meth_ripple_critics nil
succeeded

attempting atomic critic
set_goal _ _ _ _
set_goal _NewG
        _Hyp <> [(ext g) ((emb x) + H) - (emb (g x)) ≈ 0,
                    (ext g) ((emb x) + H) - (emb (g x)) = 0  ∨
                             (ext g) ((emb x) + H) - (emb (g x)) ≠ 0]

attempting atomic critic
continue_crit (M (then_meth sym_eval (then_meths or_e (pair_meth eval_meth
          (then_meth or_e nsa_top_meth_ripple_critics)))) M)
continue_crit (M (then_meth sym_eval (then_meths or_e (pair_meth eval_meth
          (then_meth or_e nsa_top_meth_ripple_critics)))) M)
succeeded
```

Notice here that there is an explicit call to the `sym_eval` method, which corresponds to discharging the subgoal

$$^*g(\widehat{x}+h)-\widehat{g(x)}\approx 0$$

which is introduced by the atomic critic `fert_spec`.

We now rejoin the action at the point where rippling fails to find a wave rule. The goal at this point now has two branches. We concentrate on the branch where $^*g(\widehat{x}+h)-\widehat{g(x)}\neq 0$.

```
h: *ℝ, x:ℝ, d1:ℝ, d2:ℝ, f:ℝ→ℝ, g:ℝ→ℝ
(ext g) (((emb x)+h) - (emb (g x))) ≈ 0
(ext g) (((emb x)+h) - (emb (g x))) ≠ 0
∀ H ∈ *ℝ. (H ≈ 0 ∧ H ≠ 0) →
      H^-1 × (ext f) ((emb (g x))+H)-(emb (f (g x)))  ≈ (emb d1)
```

```
∀ H ∈ *ℝ. (H ≈ 0 ∧ H ≠ 0) →
      H^-1 × (ext g) ((emb x)+H)-(emb (g x))  ≈ (emb d_2)
⊢
(h ≈ 0 ∧ h ≠ 0) →
(((ext g) (((emb x)+h) - (emb (g x))))^-1 ×
     (ext f) (((emb (g x))+h)-(emb (f (g x))))) ×
         ((h^-1) × (ext g) ((emb x)+h)-(emb (g x))) ≈ (emb (d_1 × d_2))
```

Now the rippling process cannot continue so the lemma speculation critic fires.

```
Attempting...
(patch_meth (wave_method outward _) wave_critic_strat)

measure_check nored outward _E _G _H
failed
.
.
.
attempting atomic critic
spec_colour_wave _ _  _ _
spec_colour_wave _ ((A × B) ≈ (C × D)) ((A ≈ C) ∧ (B ≈ D))
              ((finite B) ∧ (finite D))
```

From here the proof-plan proceeds to the point where the lemma speculation critic suggests the rule

$$A \to B \wedge C \Rightarrow A \to B \wedge A \to C$$

and then the proof-plan proceeds until the point where the goal is fully rippled out.

```
h: *ℝ, x:ℝ, d_1:ℝ, d_2:ℝ, f:ℝ→ℝ, g:ℝ→ℝ
(ext g) (((emb x)+h) - (emb (g x))) ≈ 0
(ext g) (((emb x)+h) - (emb (g x))) ≠ 0
∀ H ∈ *ℝ. (H ≈ 0 ∧ H ≠ 0) →
      H^-1 × (ext f) ((emb (g x))+H)-(emb (f (g x)))  ≈ (emb d_1)
∀ H ∈ *ℝ. (H ≈ 0 ∧ H ≠ 0) →
      H^-1 × (ext g) ((emb x)+H)-(emb (g x))  ≈ (emb d_2)
⊢
(h ≈ 0 ∧ h ≠ 0) →
   (((ext g) (((emb x)+h) - (emb (g x))))^-1 ×
          (ext f) (((emb (g x))+h)-(emb (f (g x))))) ≈ (emb d_1)    ∧
   ((h^-1) × (ext g) ((emb x)+h)-(emb (g x))) ≈ (emb d_2))
```

The `fertilise` method now succeeds since the mismatching sinks in the antecedent of the implication in the goal can be replaced with the new hypotheses added by the `fertilise` critic. We now have a complete proof-plan for the chain rule.

Figure A.2 shows the output from *XBarnacle* [Lowe and Duncan, 1997] – the graphical user interface to a previous version of λ*Clam*, in which we also constructed a proof-plan for the chain rule.

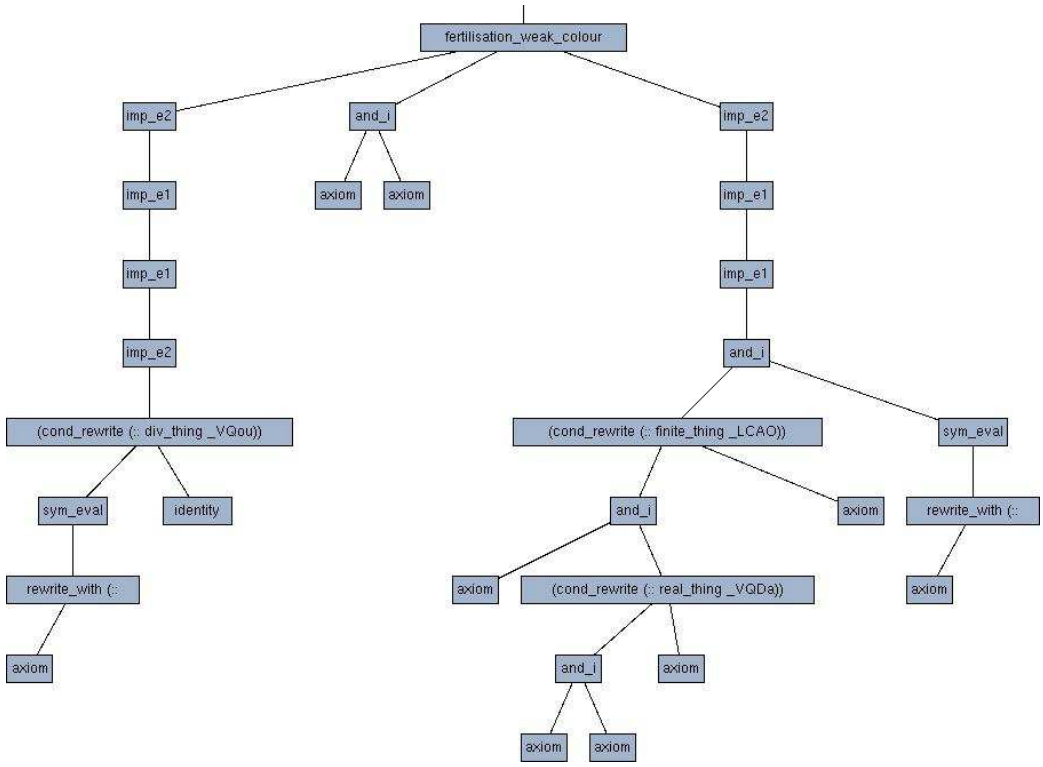

Figure A.2: The output from XBarnacle for the chain rule

A.2 Rolle's Theorem

The compound method which corresponds to the plan-specification given by figure 6.14 in section 6.5 is shown in figure A.3. We provide the plan-specification with the name of the theorem corresponding to Rolle's theorem, the names of the case-split sets to the main plan-specification shown in figure A.3. We also conjecture one inductive lemma by hand, shown in section 6.3.1 by lemma 6.13. The initial statement of Rolle's Theorem is

`fp: ℝ → ℝ, f: ℝ → ℝ, a: ℝ, b: ℝ, c: ℝ`

```
compound nsaind (nsa_partition_final Lem Rewr)
  (then_meth (collect_meth Rewr Lem Lems)
  (then_meth (induction_all Lems)
  (then_meth (generalise_all Lems Hyps)
  (then_meth (add_hyps Hyps)
  (then_meth final_steps))))).

  _
  true.
```

Figure A.3: The outermost compound method for the partitioning examples.

$$
\begin{array}{l}
\forall\ X \in {}^*\mathbb{R}.\ \forall\ H \in {}^*\mathbb{R}. \\
\quad ((\mathrm{emb}\ a) \leq X \leq (\mathrm{emb}\ b) \wedge (H \neq 0) \wedge (H \approx 0)) \rightarrow \\
\qquad H^{-1} \times (\mathrm{ext}\ f)\ (X+H)\text{-}(\mathrm{ext}\ f)\ X \approx (\mathrm{ext}\ \mathrm{fp}\ X) \\
\forall\ X \in {}^*\mathbb{R}.\ \forall\ Y \in {}^*\mathbb{R}. \\
\quad ((\mathrm{emb}\ a) \leq X \leq (\mathrm{emb}\ b) \wedge (\mathrm{emb}\ a) \leq Y \leq (\mathrm{emb}\ b) \wedge (X \approx Y)) \rightarrow \\
\qquad (\mathrm{ext}\ f)\ X \approx (\mathrm{ext}\ f)\ Y \\
b > a \\
(f\ a) = (f\ b) \\
\vdash \\
\exists\ X \in \mathbb{R}.\ (\mathrm{emb}\ a) \leq X \leq (\mathrm{emb}\ b) \rightarrow \mathrm{fp}\ X = 0
\end{array}
$$

and the inductive lemma we pass to the plan-specification is

$$
\begin{array}{l}
\mathrm{fp}\colon \mathbb{R} \rightarrow \mathbb{R},\ f\colon \mathbb{R} \rightarrow \mathbb{R},\ a\colon \mathbb{R},\ b\colon \mathbb{R},\ c\colon \mathbb{R} \\
b > a \\
(f\ a) = (f\ b) \\
\vdash \\
\forall\ N \in \mathbb{N}. \\
\quad \mathrm{fp}\ (\mathrm{rolrec}_l\ f\ \mathrm{fp}\ a\ b\ n) \geq 0 \ \wedge\ \mathrm{fp}\ (\mathrm{rolrec}_r\ f\ \mathrm{fp}\ a\ b\ n) < 0 \ \vee \\
\quad \mathrm{fp}\ (\mathrm{rolrec}_l\ f\ \mathrm{fp}\ a\ b\ n) < 0 \ \wedge\ \mathrm{fp}\ (\mathrm{rolrec}_r\ f\ \mathrm{fp}\ a\ b\ n) \geq 0 \ \vee \\
\quad \mathrm{fp}\ (\mathrm{rolrec}_l\ f\ \mathrm{fp}\ a\ b\ n) \geq 0 \ \wedge\ \mathrm{fp}\ (\mathrm{rolrec}_r\ f\ \mathrm{fp}\ a\ b\ n) \geq 0 \ \wedge \\
\qquad f\ (\mathrm{rolrec}_l\ f\ \mathrm{fp}\ a\ b\ n) \geq f\ (\mathrm{rolrec}_r\ f\ \mathrm{fp}\ a\ b\ n) \ \vee \\
\quad \mathrm{fp}\ (\mathrm{rolrec}_l\ f\ \mathrm{fp}\ a\ b\ n) < 0 \ \wedge\ \mathrm{fp}\ (\mathrm{rolrec}_r\ f\ \mathrm{fp}\ a\ b\ n) < 0 \ \wedge \\
\qquad f\ (\mathrm{rolrec}_l\ f\ \mathrm{fp}\ a\ b\ n) \leq f\ (\mathrm{rolrec}_r\ f\ \mathrm{fp}\ a\ b\ n)
\end{array}
$$

The compound method `collect_meth` finds proof-plans for all of the common inductive lemmas as well as the lemma provided to the outermost plan-specification. We give an example

output from λ*Clam* in finding a proof-plan for the theorem

$$\forall n \in \mathbb{N}.\ \texttt{rolrec}_\texttt{r}\ f\ f'\ a\ b\ n\ -\ \texttt{rolrec}_\texttt{l}\ f\ f'\ a\ b\ n = \frac{b-a}{2^n}.$$

Initially, the base case is established.

```
fp: ℝ → ℝ, f: ℝ → ℝ, a: ℝ, b: ℝ, c: ℝ
b > a
(f a) = (f b)
⊢
rolrec_r (f, fp, a, b, 0) - rolrec_l (f, fp, a, b, 0) =  (b - a)/2^0

Attempting...
Method application:  rewrite_with rol_r
succeeded
fp: ℝ → ℝ, f: ℝ → ℝ, a: ℝ, b: ℝ, c: ℝ
b > a
(f a) = (f b)
⊢
b - rolrec_l (f, fp, a, b, 0)) = (b - a)/2^0

Attempting...
Method application:  rewrite_with rol_l
succeeded
fp: ℝ → ℝ, f: ℝ → ℝ, a: ℝ, b: ℝ, c: ℝ
b > a
(f a) = (f b)
⊢
b - a = (b - a)/2^0

Attempting...
Method application:  rewrite_with exp1
succeeded
fp: ℝ → ℝ, f: ℝ → ℝ, a: ℝ, b: ℝ, c: ℝ
b > a
(f a) = (f b)
⊢
b - a = (b - a)/1
```

```
Attempting...
Method application:  rewrite_with invtimes_ident
succeeded
fp: ℝ → ℝ, f: ℝ → ℝ, a: ℝ, b: ℝ, c: ℝ
b > a
(f a) = (f b)
⊢
b - a = (b - a) × 1

Attempting...
Method application:  rewrite_with times_ident
succeeded
fp: ℝ → ℝ, f: ℝ → ℝ, a: ℝ, b: ℝ, c: ℝ
b > a
(f a) = (f b)
⊢
b - a = b - a

Attempting...
Method application:  rewrite_with refleq
succeeded
fp: ℝ → ℝ, f: ℝ → ℝ, a: ℝ, b: ℝ, c: ℝ
b > a
(f a) = (f b)
⊢ trueP

trueGoal!
branch closed!
```

Now the step case proceeds by structural induction.

```
Method application:  induction_meth nat_struct
succeeded

Method application:  step_case
succeeded

Method application:  set_up_ripple
succeeded
```

```
fp: ℝ → ℝ, f: ℝ → ℝ, a: ℝ, b: ℝ, c: ℝ
b > a
(f a) = (f b)
rolrec_r (f, fp, a, b, N) - rolrec_l (f, fp, a, b, N) = (b - a)/2^N
⊢
rolrec_r (f, fp, a, b, s N) - rolrec_l (f, fp, a, b, s N)) = (b - a)/2^s(N)
```

Once embedding has taken place, rippling can go ahead. In order to ripple using the definitions for `rolrec_l` and `rolrec_r`, we use the case split methods. We show here one branch of the case split.

```
Method application:  wave_method outward rol_r
succeeded
fp: ℝ → ℝ, f: ℝ → ℝ, a: ℝ, b: ℝ, c: ℝ
b > a
(f a) = (f b)
rolrec_r (f, fp, a, b, N) - rolrec_l (f, fp, a, b, N) = (b - a)/2^N
f((rolrec_l (f, fp, a, b, N) + rolrec_r (f, fp, a, b, N))/2) ≥
            f(rolrec_l (f, fp, a, b, N))               ∧
    fp((rolrec_l (f, fp, a, b, N) + rolrec_r (f, fp, a, b, N))/2) ≥ 0
⊢
rolrec_r (f, fp, a, b, N) - rolrec_l (f, fp, a, b, s N)) = (b - a)/2^s(N)

Method application:  wave_method outward rol_l
succeeded
fp: ℝ → ℝ, f: ℝ → ℝ, a: ℝ, b: ℝ, c: ℝ
b > a
(f a) = (f b)
rolrec_r (f, fp, a, b, N) - rolrec_l (f, fp, a, b, N) = (b - a)/2^N
f((rolrec_l (f, fp, a, b, N) + rolrec_r (f, fp, a, b, N))/2) ≥
    f(rolrec_l (f, fp, a, b, N)) ∧
        fp((rolrec_l (f, fp, a, b, N) + rolrec_r (f, fp, a, b, N))/2) ≥ 0
⊢
rolrec_r (f, fp, a, b, N) -
    (rolrec_l (f, fp, a, b, N) + rolrec_r (f, fp, a, b, N))/2  =
            (b - a)/2^s(N)
```

Now the lemma speculation critic fires and speculates the rule which allows fertilisation to take place.

After the transfer plan-specification is successfully applied, the statement of Rolle's theorem is augmented with new hypotheses. Here θ represents the ${}^*l(n)$ term introduced by the transfer plan-specification, and ϕ represents ${}^*r(n)$.

```
fp: ℝ → ℝ, f: ℝ → ℝ, a: ℝ, b: ℝ, c: ℝ, θ: *ℝ, ϕ: *ℝ
θ ≈ ϕ
ϕ > θ
(emb a) ≤ θ ≤ (emb b)
(emb a) ≤ ϕ ≤ (emb b)
(ext f) θ ≈ (ext f) ϕ
(ext fp) θ ≈ (ext fp) ϕ
((ext fp) θ ≥ 0 ∧ (ext fp) ϕ <0) ∨
 ((ext fp) θ < 0 ∧ (ext fp) ϕ < 0 ∧ (ext f) θ ≤ (ext f) ϕ) ∨
  ((ext fp) θ <0 ∧ (ext fp) ϕ ≥ 0) ∨
   ((ext fp) θ ≥ 0 ∧ (ext fp) ϕ ≥ 0 ∧ (ext f) θ ≥ (ext f) ϕ
∀ X ∈ *ℝ. ∀ H ∈ *ℝ.
  ((emb a) ≤ X ≤ (emb b) ∧ (H ≠ 0) ∧ (H ≈ 0)) →
      H⁻¹ × (ext f) (X+H)-(ext f) X  ≈ (ext fp X)
∀ X ∈ *ℝ. ∀ Y ∈ *ℝ.
  ((emb a) ≤ X ≤ (emb b) ∧ (emb a) ≤ Y ≤ (emb b) ∧ (X ≈ Y)) →
     (ext f) X ≈ (ext f) Y
b > a
(f a) = (f b)
⊢
∃ X ∈ ℝ. (emb a) ≤ X ≤ (emb b) →
   fp X = 0
```

Now the final part of the plan-specification rewrites the hypotheses. The proof-plan follows that shown in section 6.3.1 using the compound method of figure A.4. This method describes the transfer-back plan-specification of figure 6.14 in section 6.5.3. The final stages of the proof-plan yield new hypotheses which we can use to discharge the goal. After application of method `disc_cond` we yield the goal

```
fp: ℝ → ℝ, f: ℝ → ℝ, a: ℝ, b: ℝ, c: ℝ, θ: *ℝ, ϕ: *ℝ, x: ℝ
θ ≈ ϕ
ϕ > θ
(emb a) ≤ θ ≤ (emb b)
(emb a) ≤ ϕ ≤ (emb b)
(ext f) θ ≈ (ext f) ϕ
(ext fp) θ ≈ (ext fp) ϕ
```

```
compound nsaind final_steps Goal
  (then_meth disc_cond
  (then_meth int_witness
  (then_meth
  (repeat_meth
    est_bounds)
  (then_meth int_cases
  (then_meth simp_deriv
  (then_meth ord_const
    (repeat_meth realise_wit)))))))
  _
  true.
```

Figure A.4: The compound method for the final part of the partitioning examples

$$
\begin{array}{l}
((\text{ext fp})\ \theta \geq 0 \wedge (\text{ext fp})\ \phi < 0) \vee \\
\ ((\text{ext fp})\ \theta < 0 \wedge (\text{ext fp})\ \phi < 0 \wedge (\text{ext f})\ \theta \leq (\text{ext f})\ \phi) \vee \\
\ \ ((\text{ext fp})\ \theta < 0 \wedge (\text{ext fp})\ \phi \geq 0) \vee \\
\ \ \ ((\text{ext fp})\ \theta \geq 0 \wedge (\text{ext fp})\ \phi \geq 0 \wedge (\text{ext f})\ \theta \geq (\text{ext f})\ \phi \\
\forall\ \text{X} \in {}^*\mathbb{R}.\ \forall\ \text{H} \in {}^*\mathbb{R}. \\
\ \ ((\text{emb a}) \leq \text{X} \leq (\text{emb b}) \wedge (\text{H} \neq 0) \wedge (\text{H} \approx 0)) \rightarrow \\
\ \ \ \ \text{H}^{-1} \times (\text{ext f})\ (\text{X+H})\text{-}(\text{ext f})\ \text{X} \ \approx (\text{ext fp X}) \\
\forall\ X \in {}^*\mathbb{R}.\ \forall\ \text{Y} \in {}^*\mathbb{R}. \\
\ \ ((\text{emb a}) \leq \text{X} \leq (\text{emb b}) \wedge (\text{emb a}) \leq \text{Y} \leq (\text{emb b}) \wedge (\text{X} \approx \text{Y})) \rightarrow \\
\ \ \ \ (\text{ext f})\ \text{X} \approx (\text{ext f})\ \text{Y} \\
\text{b} > \text{a} \\
(\text{f a}) = (\text{f b}) \\
\vdash \\
\exists\ \text{X} \in \mathbb{R}.\ (\text{emb a}) \leq \text{X} \leq (\text{emb b}) \rightarrow \\
\ \ \text{fp X} = 0
\end{array}
$$

Now when the witness has been introduced by the method `int_witness`, and the bounds have been added by method `est_bounds`, the planner uses method `int_cases` to introduce four cases. We consider one case here:

$$
\begin{array}{l}
\text{fp}: \mathbb{R} \rightarrow \mathbb{R},\ \text{f}: \mathbb{R} \rightarrow \mathbb{R},\ \text{a}: \mathbb{R},\ \text{b}: \mathbb{R},\ \text{c}: \mathbb{R},\ \theta: {}^*\mathbb{R},\ \phi: {}^*\mathbb{R},\ \text{x}: \mathbb{R} \\
\theta \approx \phi
\end{array}
$$

```
ϕ > θ
(emb a) ≤ θ ≤ (emb b)
(emb a) ≤ ϕ ≤ (emb b)
(ext f) θ ≈ (ext f) ϕ
(ext fp) θ ≈ (ext fp) ϕ
((ext fp) θ ≥ 0 ∧ (ext fp) ϕ <0)
∀ X ∈ *ℝ. ∀ H ∈ *ℝ.
  ((emb a) ≤ X ≤ (emb b) ∧ (H ≠ 0) ∧ (H ≈ 0)) →
      H⁻¹ × (ext f) (X+H)-(ext f) X  ≈ (ext fp X)
∀ X ∈ *ℝ. ∀ Y ∈ *ℝ.
  ((emb a) ≤ X ≤ (emb b) ∧ (emb a) ≤ Y ≤ (emb b) ∧ (X ≈ Y)) →
     (ext f) X ≈ (ext f) Y
b > a
(f a) = (f b)
(emb x) ≈ θ
a ≤ x ≤ b
⊢
∃ X ∈ ℝ. (emb a) ≤ X ≤ (emb b) →
   fp X = 0
```

Now methods `simp_deriv` and `ord_const` apply and we yield the goal

```
fp: ℝ → ℝ, f: ℝ → ℝ, a: ℝ, b: ℝ, c: ℝ, θ: *ℝ, ϕ: *ℝ, x: ℝ
θ ≈ ϕ
ϕ > θ
(emb a) ≤ θ ≤ (emb b)
(emb a) ≤ ϕ ≤ (emb b)
(ext f) θ ≈ (ext f) ϕ
(ext fp) θ ≈ (ext fp) ϕ
((ext fp) θ ≥ 0 ∧ (ext fp) ϕ <0)
∀ X ∈ *ℝ. ∀ H ∈ *ℝ.
  ((emb a) ≤ X ≤ (emb b) ∧ (H ≠ 0) ∧ (H ≈ 0)) →
      H⁻¹ × (ext f) (X+H)-(ext f) X  ≈ (ext fp X)
∀ X ∈ *ℝ. ∀ Y ∈ *ℝ.
  ((emb a) ≤ X ≤ (emb b) ∧ (emb a) ≤ Y ≤ (emb b) ∧ (X ≈ Y)) →
     (ext f) X ≈ (ext f) Y
b > a
(f a) = (f b)
(emb x) ≈ θ
```

```
a ≤ x ≤ b
(ext fp) θ ≈ 0
⊢
∃ X ∈ ℝ. (emb a) ≤ X ≤ (emb b) →
   fp X = 0
```

Method `realise_wit` now applies and the existentially quantified variable in the conclusion is instantiated to `(emb x)`, since `fp x = 0` can be established.

A.3 Example code for atomic methods

We give some simple example code for the methods introduced in chapters 5 and 6. Method (3 shown in section 5.4.4 of chapter 5 is entered as follows:

```
atomic nsaconjectures inf_int
     (seqGoal (H >>> G))
     (memb (app eq_ic (tuple [Varl,Varr])) H,
     T_1 = (app eq_ic (tuple [Int_var,zero])),
     T_2 = (app eq (tuple [app minus (tuple [Var1,Var2],Int_var)])),
     replace_vars (app plus (tuple [Varr,Int_var])) Varl G Gnew)
     true
     (seqGoal ((T_1 :: T_2 :: H) >>> Gnew))
     notacticyet.
```

The auxiliary predicate `replace_vars` is written

```
%%% replace_vars/4 + + + -
%%% instantiate #4 with #3 with all occurrences of #2 replaced with #1

replace_vars Varout Varin X Varout :-
        not (headvar_osyn X),
        !,
        X = Varin.

replace_vars Varout Varin X X :-
        headvar_osyn X,
        !.

replace_vars Varout Varin X X:-
        not (headvar_osyn X),
```

```
        X = (otype_of _ _).

replace_vars Varout Varin X (app C D):-
        not (headvar_osyn X),
        X = (app A B),
        !,
        replace_vars Varout Varin A C,
        replace_vars Varout Varin B D.

replace_vars Varout Varin X (tuple Lout):-
        not (headvar_osyn X),
        X = (tuple Lin),
        !,
        rec_replace_vars Varout Varin Lin Lout.

replace_vars Varout Varin X (abs Y) :-
        not (headvar_osyn X),
        X = (abs A),
        !,
        pi u (replace_vars Varout Varin (A u) (Y u)).

%%% Recursive auxiliary function to replace_vars for tuple case
rec_replace_vars _Varout _Varin nil nil.

rec_replace_vars Varout Varin (H::T) (H1::T1) :-
        replace_vars Varout Varin H H1,
        rec_replace_vars Varout Varin T T1.
```

The discharge conditions method shown in section 6.6.3 of chapter 6 is entered as follows:

```
atomic nsaind disc_gen_hyp
     (seqGoal (H >>> G))
     (memb (otype_of Var1 hyperreal) H,
     memb (otype_of Var2 hyperreal) H,
     memb (app eq_ic (tuple [Var1,Var2])) H,
     memb (app neg (app eq (tuple [Var1,Var2]))) H,
     findall (Limp Land Rand (memb (app imp (tuple [Limp, (app and
                    (tuple [Land,Rand]))])) H)) Imp_args,
     forthose Imp_args (I V1 V2 disc_match I V1 V2 List) Var1 Var2,
```

```
append H List NewH)
true
(seqGoal (NewH >>> G))
notacticyet.
```

Appendix B

Proof-plans for the inductive lemmas

We describe the proofs that correspond to the proof-plans that $\lambda Clam$ constructed for the Intermediate Value Theorem, and for the inductive lemmas for Rolle's Theorem. We also present the lemmas which were added to $\lambda Clam$ interactively for the theorems presented in chapter 6.

B.1 Proof of the Intermediate Value Theorem

Recall first the definition of the intermediate value theorem

$$f : \mathbb{R} \to \mathbb{R}$$

$$a, b, c : \mathbb{R}$$

$$\forall x, y \in {}^*\mathbb{R}.\ \widehat{a} \leq x \leq \widehat{b} \wedge \widehat{a} \leq y \leq \widehat{b} \wedge x \approx y \ \rightarrow\ {}^*f(x) \approx {}^*f(y)$$

$$a < b$$

$$f(a) \geq c \geq f(b)$$

$$\vdash \exists x \in \mathbb{R}.\ a \leq x \leq b \wedge f(x) = c.$$

Firstly we state the partitioning function

```
ivtrec F A B C 0 = [A,B]

ivtrec F A B C s(N) =  (let [X,Y]=ivtrec F A B C N
                          in if F((X+Y)/2)>C then [(X+Y)/2,Y]
                                             else [X,(X+Y)/2]).
```

In the implementation we represent the partitioning function by rewrite rules, and specify one rule for the left point of the interval, and one for the right point of the interval. So we define

rules ivtrec_r and ivtrec_l as rewrite rules, and annotate them to become wave rules as shown in figure 6.3. Now we can use these wave rules to prove important conjectures about the partitioning function. Recall from section 6.2 that we need to prove the following theorems

$$\forall n \in {}^*\mathbb{N}.\ {}^*\mathtt{ivtrec_r}\ {}^*f\ \widehat{a}\ \widehat{b}\ \widehat{c}\ n\ -\ {}^*\mathtt{ivtrec_l}\ {}^*f\ \widehat{a}\ \widehat{b}\ \widehat{c}\ n = \tfrac{\widehat{b}-\widehat{a}}{2^n}$$
$$\forall n \in {}^*\mathbb{N}.\ {}^*\mathtt{ivtrec_r}\ {}^*f\ \widehat{a}\ \widehat{b}\ \widehat{c}\ n\ \geq a \wedge \forall n \in {}^*\mathbb{N}.\ {}^*\mathtt{ivtrec_r}\ {}^*f\ \widehat{a}\ \widehat{b}\ \widehat{c}\ n\ \leq b$$
$$\forall n \in {}^*\mathbb{N}.\ {}^*\mathtt{ivtrec_l}\ {}^*f\ \widehat{a}\ \widehat{b}\ \widehat{c}\ n\ \geq a \wedge \forall n \in {}^*\mathbb{N}.\ {}^*\mathtt{ivtrec_l}\ {}^*f\ \widehat{a}\ \widehat{b}\ \widehat{c}\ n\ \leq b$$
$$\forall n \in {}^*\mathbb{N}.\ {}^*f({}^*\mathtt{ivtrec_r}\ {}^*f\ \widehat{a}\ \widehat{b}\ \widehat{c}\ n)\ \geq\ \widehat{c} \wedge\ \widehat{c}\ \geq\ {}^*f({}^*\mathtt{ivtrec_l}\ {}^*f\ \widehat{a}\ \widehat{b}\ \widehat{c}\ n).$$

Once the proofs of these theorems have been planned, we generalise them by hand, by substituting ${}^*l(n)$ for ${}^*\mathtt{ivtrec_l}\ {}^*f\ \widehat{a}\ \widehat{b}\ \widehat{c}\ n$, and ${}^*r(n)$ for ${}^*\mathtt{ivtrec_r}\ {}^*f\ \widehat{a}\ \widehat{b}\ \widehat{c}\ n$. We can then add the generalised facts to the hypotheses to yield the new statement for the intermediate value theorem given by 6.7. To show all of the proofs of theorems planned, we first show the proof of the intermediate theorem using the new generalisations, so that we justify the use of the inductive conjectures whose proofs we show after.

In order to prove the intermediate value theorem we must satisfy the existentially quantified conclusion

$$\exists x : \mathbb{R}.\ a \leq x \leq b \rightarrow f(x) = c$$

with the hypotheses

$$n : {}^*\mathbb{R} \tag{B.1}$$
$$f : \mathbb{R} \rightarrow \mathbb{R} \tag{B.2}$$
$$a, b, c : \mathbb{R} \tag{B.3}$$
$$\neg\, finite(n) \tag{B.4}$$
$${}^*l(n) \approx {}^*r(n) \tag{B.5}$$
$$\widehat{a} \leq {}^*r(n) \leq \widehat{b} \tag{B.6}$$
$$\widehat{a} \leq {}^*l(n) \leq \widehat{b} \tag{B.7}$$
$${}^*f({}^*l(n)) \geq \widehat{c} \geq {}^*f({}^*r(n)) \tag{B.8}$$
$$\forall x, y \in {}^*\mathbb{R}.\ \widehat{a} \leq x \leq \widehat{b} \wedge \widehat{a} \leq y \leq \widehat{b} \wedge x \approx y\ \rightarrow\ {}^*f(x) \approx {}^*f(y) \tag{B.9}$$
$$a < b \tag{B.10}$$
$$f(a) \geq c \geq f(b). \tag{B.11}$$

Now we use uniform continuity given (B.9) together with the facts we now about ${}^*l(n)$ and ${}^*r(n)$ given by hypotheses (B.6), (B.7) and (B.5) to deduce that

$$f({}^*l(n)) \approx f({}^*r(n)).$$

From this we can use facts we know from the axiomatisation to find a real number which satisfies the conclusion given by (B.1). We know from axiom (4.38) that for any finite hyperreal, there is a real number which lies infinitely close. From axiom (4.40) we can infer that this number is unique. We can also infer that there a unique real in the infinitesimal neighbourhood. In order to establish the theorem we must show that this number lies between a and b, and that its image under the function f is at c. We proceed by stating a number of lemmas.

$\widehat{x} \approx {}^*l(n)$

We can state this directly from axiom (4.38). This ensures we know the existence of a real number in the infinitesimal neighbourhood of any interval in the sequence produced by the partitioning function.

$a \leq x \leq b$

We now state this lemma to ascertain that the existential variable x which satisfies the previous lemma now lies within the interval. We state the theorem as:

$$x, a, b : \mathbb{R}$$
$$n : {}^*\mathbb{N}$$
$$\widehat{a} \leq {}^*l(n) \leq \widehat{b}$$
$$\widehat{x} \approx {}^*l(n) \vdash$$
$$a \leq x \leq b$$

We split the hypothesis $\widehat{a} \leq {}^*l(n)$ into to goals. In the case where $\widehat{a} = {}^*l(n)$ we yield $x = a$ by transitivity. In the case where $\widehat{a} < {}^*l(n)$, we use the lemma given in section 6.6.2 to determine that

$$\widehat{x} \approx \widehat{a} \vee \widehat{x} > \widehat{a}$$

which allows us to ascertain that $a \leq x$. We perform a similar pattern of proof for the branch where ${}^*l(n) \leq \widehat{b}$ and we prove $x \leq b$.

${}^* f({}^*l(n)) \approx \widehat{c}$

This subgoal follows by noticing that c lies between ${}^*f({}^*l(n))$ and ${}^*f({}^*r(n))$, and that ${}^*l(n) \approx {}^*r(n)$. We write the following theorem

$$n : {}^*\mathbb{N}$$
$$f : \mathbb{R} \to \mathbb{R}$$
$$a, b, c : \mathbb{R}$$
$$\neg finite(n)$$
$$\widehat{a} \leq {}^*r(n) \leq \widehat{b}$$
$$\widehat{a} \leq {}^*l(n) \leq \widehat{b}$$
$${}^*f({}^*r(n)) \geq \widehat{c} \geq {}^*f({}^*r(n))$$
$$\forall x, y \in {}^*\mathbb{R}.\ \widehat{a} \leq x \leq \widehat{b} \wedge \widehat{a} \leq y \leq \widehat{b} \wedge x \approx y \ \rightarrow\ {}^*f(x) \approx {}^*f(y)$$
$$a < b$$
$$f(a) \geq c \geq f(b) \vdash$$
$${}^*f({}^*l(n)) \approx \widehat{c}.$$

Firstly we use continuity (B.12) from the hypotheses to write the fact

$${}^*f({}^*l(n)) \approx {}^*f({}^*r(n)).$$

Now we also notice that

$${}^*f({}^*l(n)) \geq \widehat{c} \geq {}^*f({}^*r(n)).$$

We introduce the following rules:

$$A \approx B \wedge A < C \wedge B > C \quad \rightarrow \quad A \approx C$$
$$A \approx B \wedge A < C \wedge B > C \quad \rightarrow \quad B \approx C.$$

In the case where $f({}^*l(n)) = \widehat{c}$ we use rule (4.31) to show that $f({}^*l(n)) \approx \widehat{c}$. In the case where $f({}^*l(n)) > \widehat{c}$ we use the above rules to obtain the result.

$f(x) = c$

Using the lemmas we have already planned we can form the lemma

$$n : {}^*\mathbb{N}$$
$$f : \mathbb{R} \to \mathbb{R}$$

$$
\begin{gathered}
a,b,c,x:\mathbb{R} \\
\neg finite(n) \\
\forall x,y \in {}^*\mathbb{R}.\ \widehat{a} \leq x \leq \widehat{b} \wedge \widehat{a} \leq y \leq \widehat{b} \wedge x \approx y \rightarrow {}^*f(x) \approx {}^*f(y) \\
\widehat{x} \approx {}^*l(n) \\
{}^*f({}^*l(n)) \approx \widehat{c} \vdash \\
f(x) = c.
\end{gathered}
\tag{B.12}
$$

We can immediately use continuity (B.12) to determine that

$$\widehat{f(x)} \approx {}^*f({}^*l(n))$$

and then transitivity with (B.12) to see that

$$\widehat{f(x)} \approx \widehat{c}$$

Now from axiom (4.40) we can determine that $f(x) = c$ as required.

Now it just remains to present proofs of the inductive theorems. Here we present a proof-plan for the inductive proof of each of the theorems we proved above. We use the internal representation of rewrite rules, and separate partitioning functions for the left and the right end point of each partition. In the presentations of the inductive proofs that follow we express $\geq$ explicitly as $> \vee =$, likewise for $\leq$.

Lemma (6.3)

We write this theorem as

$$\forall n \in \mathbb{N}.\ \texttt{ivtrec}_\texttt{r}\ f\ a\ b\ c\ n\ -\ \texttt{ivtrec}_\texttt{l}\ f\ a\ b\ c\ n = \frac{b-a}{2^n}.$$

We then use a standard structural induction scheme for the natural numbers, and choose single-step induction on n– the only candidate for induction. The resulting non-standard formulation is

$$\forall n \in {}^*\mathbb{N}.\ {}^*\texttt{ivtrec}_\texttt{r}\ {}^*f\ \widehat{a}\ \widehat{b}\ \widehat{c}\ n\ -\ {}^*\texttt{ivtrec}_\texttt{l}\ {}^*f\ \widehat{a}\ \widehat{b}\ \widehat{c}\ n = \frac{\widehat{b}-\widehat{a}}{2^n}.$$

We use this to show that for an infinite n, the interval defined by `ivtrec` is infinitely small. The base case to the proof of the lemma is

$$\texttt{ivtrec}_\texttt{r}\ f\ a\ b\ c\ 0 - \texttt{ivtrec}_\texttt{l}\ f\ a\ b\ c\ 0 = b - a$$

which is trivially proved by rewriting and identity, using the wave-rules defined for the partitioning function, the definition for exponentiation given by axiom (4.6) and the field equations for multiplication (4.15), (4.24) and (4.16).

$$\texttt{ivtrec}_\texttt{r}\ f\ a\ b\ c\ 0 - \texttt{ivtrec}_\texttt{l}\ f\ a\ b\ c\ 0 = \frac{b-a}{2^0}$$

can be rewritten to

$$b-a = \frac{b-a}{2^0}$$

which can be rewritten to

$$b-a = b-a$$

completing the proof of the base case. The step case becomes

$$\texttt{ivtrec}_\texttt{r}\ f\ a\ b\ c\ \boxed{s(n)}^{\uparrow} - \texttt{ivtrec}_\texttt{l}\ f\ a\ b\ c\ \boxed{s(n)}^{\uparrow} = \frac{b-a}{2^{\boxed{s(n)}^{\uparrow}}}.$$

At this point we introduce two cases, which correspond to the conditions to the wave rules for $\texttt{ivtrec}_\texttt{r}$ and $\texttt{ivtrec}_\texttt{l}$. We deal first with the case where

$$f((\texttt{ivtrec}_\texttt{l}\ f\ a\ b\ c\ n + \texttt{ivtrec}_\texttt{r}\ f\ a\ b\ c\ n)/2) > c.$$

After the application of arithmetical rules (4.9) and (4.15), the conclusion becomes

$$\boxed{\frac{\texttt{ivtrec}_\texttt{l}\ f\ a\ b\ c\ n + \boxed{\texttt{ivtrec}_\texttt{r}\ f\ a\ b\ c\ n}}{2}}^{\uparrow} - \texttt{ivtrec}_\texttt{l}\ f\ a\ b\ c\ n = \boxed{\frac{1}{2} \times \frac{b-a}{2^n}}^{\uparrow}.$$

Now we speculate the wave rule

$$\boxed{\frac{Y + \boxed{X}}{2}}^{\uparrow} - Y \quad \Rightarrow \quad \boxed{\frac{1}{2} \times \boxed{X-Y}}^{\uparrow}$$

which can now be used to rewrite the conclusion to

$$\boxed{\frac{1}{2} \times \boxed{\texttt{ivtrec}_\texttt{r}\ f\ a\ b\ c\ n - \texttt{ivtrec}_\texttt{l}\ f\ a\ b\ c\ n}}^{\uparrow} = \boxed{\frac{1}{2} \times \frac{b-a}{2^n}}^{\uparrow}.$$

Now the proof is completed by rewriting with the induction hypothesis, using weak fertilisiation. At this point in the proof, the natural step to take is to cancel the $\frac{1}{2}$ terms on either side of the equality. In proof-planning terminology, this would allow strong fertilisation to take place- i.e. a direct invocation of the hypothesis. In our case we use weak fertilisation by using the substitution rule we include in our logic, and then

complete the proof using the reflexivity of equality. The resulting goal is an identity, which is proved using the tautology method.

For the other case, we add to the hypotheses the fact

$$f((\texttt{ivtrec}_\texttt{l}\ f\ a\ b\ c\ n + \texttt{ivtrec}_\texttt{r}\ f\ a\ b\ c\ n)/2) < c \vee$$
$$f((\texttt{ivtrec}_\texttt{l}\ f\ a\ b\ c\ n + \texttt{ivtrec}_\texttt{r}\ f\ a\ b\ c\ n)/2) = c.$$

We can reason in a similar manner to the other case. We first ripple using the rules defined for $\texttt{ivtrec}_\texttt{l}$ and $\texttt{ivtrec}_\texttt{r}$, and reach the annotated conclusion

$$\texttt{ivtrec}_\texttt{r}\ f\ a\ b\ c\ n - \frac{\texttt{ivtrec}_\texttt{l}\ f\ a\ b\ c\ n + \texttt{ivtrec}_\texttt{r}\ f\ a\ b\ c\ n}{2}{}^{\uparrow} = \frac{1}{2} \times \frac{b-a}{2^n}{}^{\uparrow}$$

for which we speculate the following wave rule

$$X - \frac{X+Y}{2}{}^{\uparrow} \Rightarrow X - Y{}^{\uparrow} 2 \tag{B.13}$$

which allows us to rewrite the conclusion to the point where we can employ weak fertilisation as in the first case of the proof.

Lemmas (6.4) and (6.5)

We write these lemmas together as

$$\forall n \in \mathbb{N}.\ \texttt{ivtrec}_\texttt{r}\ f\ a\ b\ c\ n \geq a \wedge \texttt{ivtrec}_\texttt{r}\ f\ a\ b\ c\ n \leq b\ \wedge$$
$$\texttt{ivtrec}_\texttt{l}\ f\ a\ b\ c\ n \geq a \wedge \texttt{ivtrec}_\texttt{l}\ f\ a\ b\ c\ n \leq b$$

In order to prove this goal however, we must use mutual induction. We perform a case split on the conditions to the recursive wave rule definition for $\texttt{ivtrec}_\texttt{l}$ and $\texttt{ivtrec}_\texttt{r}$. Let us first state the conjectures which require proof.

We state the conjecture for the left boundedness of the leftmost point of the partition.

$$\forall n \in \mathbb{N}.\ \texttt{ivtrec}_\texttt{l}\ f\ a\ b\ c\ n > a \vee \texttt{ivtrec}_\texttt{l}\ f\ a\ b\ c\ n = a$$

for which we first state the base case

$$\texttt{ivtrec}_\texttt{l}\ f\ a\ b\ c\ n > a \vee \texttt{ivtrec}_\texttt{l}\ f\ a\ b\ c\ n = a$$

which is rewritten immediately using the wave rules introduced to

$$a > a \vee a = a$$

which is proved using the tautology method. The right boundedness condition is written

$$\forall n \in \mathbb{N}.\ \texttt{ivtrec}_\texttt{l}\ f\ a\ b\ c\ n < b \vee \texttt{ivtrec}_\texttt{l}\ f\ a\ b\ c\ n = b$$

for which the base case can be rewritten to

$$a < b \vee a = b$$

which is also proved by the tautology method.

We next state the conjecture for the left boundedness of the rightmost point of the partition.

$$\forall n \in \mathbb{N}.\ \texttt{ivtrec}_\texttt{r}\ f\ a\ b\ c\ n > a \vee \texttt{ivtrec}_\texttt{r}\ f\ a\ b\ c\ n = a$$

for which we state the base case

$$\texttt{ivtrec}_\texttt{r}\ f\ a\ b\ c\ 0 > a \vee \texttt{ivtrec}_\texttt{r}\ f\ a\ b\ c\ 0 = a$$

which can be rewritten to

$$b > a \vee b = a$$

which is proved by the tautology method. The right boundedness condition is written:

$$\forall n \in \mathbb{N}.\ \texttt{ivtrec}_\texttt{r}\ f\ a\ b\ c\ n < b \vee \texttt{ivtrec}_\texttt{r}\ f\ a\ b\ c\ n = b$$

for which we state the base case:

$$\texttt{ivtrec}_\texttt{r}\ f\ a\ b\ c\ 0 < b \vee \texttt{ivtrec}_\texttt{r}\ f\ a\ b\ c\ 0 = b$$

which can be rewritten to

$$b > b \vee b = b$$

which is proved by the tautology method.

The step case conclusion for the left boundedness of the left point of the partition is written

$$\texttt{ivtrec}_\texttt{l}\ f\ a\ b\ c\ s(n)^{\uparrow} > a \vee \texttt{ivtrec}_\texttt{l}\ f\ a\ b\ c\ s(n)^{\uparrow} = a$$

and for the right point,

$$\texttt{ivtrec}_\texttt{r}\ f\ a\ b\ c\ \boxed{s(n)}^{\uparrow} > a \vee \texttt{ivtrec}_\texttt{r}\ f\ a\ b\ c\ \boxed{s(n)}^{\uparrow} = a.$$

Now for the right boundedness for the left point the step case conclusion is written

$$\texttt{ivtrec}_\texttt{l}\ f\ a\ b\ c\ \boxed{s(n)}^{\uparrow} < b \vee \texttt{ivtrec}_\texttt{l}\ f\ a\ b\ c\ \boxed{s(n)}^{\uparrow} = b$$

and for the right point

$$\texttt{ivtrec}_\texttt{r}\ f\ a\ b\ c\ \boxed{s(n)}^{\uparrow} < b \vee \texttt{ivtrec}_\texttt{r}\ f\ a\ b\ c\ \boxed{s(n)}^{\uparrow} = b.$$

Now we construct cases according to the conditions attached to the recursive wave rules for $\texttt{ivtrec}_\texttt{l}$ and $\texttt{ivtrec}_\texttt{r}$. The first case we consider is

$$f(\frac{(\texttt{ivtrec}_\texttt{l}\ f\ a\ b\ c\ n + \texttt{ivtrec}_\texttt{r}\ f\ a\ b\ c\ n)}{2}) > c.$$

For this case, we can rewrite two of these four conclusions immediately to

$$\texttt{ivtrec}_\texttt{l}\ \texttt{f}\ \texttt{a}\ \texttt{b}\ \texttt{c}\ \texttt{n} > a \vee \texttt{ivtrec}_\texttt{l}\ \texttt{f}\ \texttt{a}\ \texttt{b}\ \texttt{c}\ \texttt{n} = a$$

$$\texttt{ivtrec}_\texttt{l}\ \texttt{f}\ \texttt{a}\ \texttt{b}\ \texttt{c}\ \texttt{n} < b \vee \texttt{ivtrec}_\texttt{l}\ \texttt{f}\ \texttt{a}\ \texttt{b}\ \texttt{c}\ \texttt{n} = b$$

which can be solved as it corresponds to the induction hypothesis.

In the following presentation, the terms in the red wave holes correspond to the induction hypotheses for $\texttt{ivtrec}_\texttt{l}$, and the terms in the blue wave holes correspond to the induction hypothesis for $\texttt{ivtrec}_\texttt{r}$. The goal is

$$\boxed{\frac{\texttt{ivtrec}_\texttt{l}\ f\ a\ b\ c\ n + \texttt{ivtrec}_\texttt{r}\ f\ a\ b\ c\ n}{2}}^{\uparrow} < b \vee \boxed{\frac{\texttt{ivtrec}_\texttt{l}\ f\ a\ b\ c\ n + \texttt{ivtrec}_\texttt{r}\ f\ a\ b\ c\ n}{2}}^{\uparrow} = b.$$

We use the lemmas

$$\boxed{\frac{X+Y}{2}}^{\uparrow} > Z \vee \boxed{\frac{X+Y}{2}}^{\uparrow} = Z \Rightarrow \boxed{X > Z \vee X = Z} \wedge \boxed{Y > Z \vee Y = Z}^{\uparrow}$$

$$\boxed{\frac{X+Y}{2}}^{\uparrow} < Z \vee \boxed{\frac{X+Y}{2}}^{\uparrow} = Z \Rightarrow \boxed{X < Z \vee X = Z} \wedge \boxed{Y < Z \vee Y = Z}^{\uparrow}$$

by which we can rewrite the conclusion to

$$\boxed{\texttt{ivtrec}_\texttt{l}\ f\ a\ b\ c\ n < b \vee \texttt{ivtrec}_\texttt{l}\ f\ a\ b\ c\ n = b} \wedge \boxed{\texttt{ivtrec}_\texttt{r}\ f\ a\ b\ c\ n < b \vee \texttt{ivtrec}_\texttt{r}\ f\ a\ b\ c\ n = b}^{\uparrow}.$$

We can perform the same proof steps for the left boundedness conjecture and arrive at the conclusion

$$\boxed{\texttt{ivtrec}_\texttt{l}\ f\ a\ b\ c\ n > a \vee \texttt{ivtrec}_\texttt{l}\ f\ a\ b\ c\ n = a} \wedge \boxed{\texttt{ivtrec}_\texttt{r}\ f\ a\ b\ c\ n > a \vee \texttt{ivtrec}_\texttt{r}\ f\ a\ b\ c\ n = a}^{\uparrow}.$$

Now the subgoals are all proved for this branch of the case split, as strong fertilisation can now apply. The other cases of the proof-plan proceed similarly.

Lemma (6.6)

We write this as two lemmas

$$\forall n \in \mathbb{N}.\ f(\mathtt{ivtrec_r}\ f\ a\ b\ c\ n) > c \vee f(\mathtt{ivtrec_r}\ f\ a\ b\ c\ n) = c$$
$$\forall n \in \mathbb{N}.\ f(\mathtt{ivtrec_l}\ f\ a\ b\ c\ n) < c \vee f(\mathtt{ivtrec_l}\ f\ a\ b\ c\ n) = c$$

which can also be solved using induction. The base cases becomes

$$f(\mathtt{ivtrec_r}\ f\ a\ b\ c\ 0) > c \vee f(\mathtt{ivtrec_r}\ f\ a\ b\ c\ 0) = c$$

and

$$f(\mathtt{ivtrec_l}\ f\ a\ b\ c\ 0) < c \vee f(\mathtt{ivtrec_l}\ f\ a\ b\ c\ 0) = c$$

which reduce to

$$f(b) > c \vee f(b) = c$$

and

$$f(a) < c \vee f(a) = c$$

which can be proved using the tautology method.

The annotated step case conclusions become

$$f(\mathtt{ivtrec_r}\ f\ a\ b\ c\ s(n)^{\uparrow}) > c \vee f(\mathtt{ivtrec_r}\ f\ a\ b\ c\ s(n)^{\uparrow}) = c$$

and

$$f(\mathtt{ivtrec_l}\ f\ a\ b\ c\ s(n)^{\uparrow}) < c \vee f(\mathtt{ivtrec_l}\ f\ a\ b\ c\ s(n)^{\uparrow}) = c$$

which produce two separate proofs each with two cases.

First we consider the case where

$$f(\frac{\mathtt{ivtrec_l}\ f\ a\ b\ c\ n + \mathtt{ivtrec_r}\ f\ a\ b\ c\ n}{2}) < c \vee f(\frac{\mathtt{ivtrec_l}\ f\ a\ b\ c\ n + \mathtt{ivtrec_r}\ f\ a\ b\ c\ n}{2}) = c.$$

The first conclusion then becomes

$$f(\mathtt{ivtrec_r\ f\ a\ b\ c\ n}) > c \vee f(\mathtt{ivtrec_r\ f\ a\ b\ c\ n}) = c$$

which corresponds to the induction hypothesis, and hence strong fertilisation applies. The second conclusion for this case becomes

$$f(\frac{\mathtt{ivtrec_l\ f\ a\ b\ c\ n} + \mathtt{ivtrec_r\ f\ a\ b\ c\ n}}{2}^{\uparrow}) < c \vee f(\frac{\mathtt{ivtrec_l\ f\ a\ b\ c\ n} + \mathtt{ivtrec_r\ f\ a\ b\ c\ n}}{2}^{\uparrow}) = c$$

which corresponds exactly to the condition add through the case split.

The next case introduces the hypothesis

$$f(\frac{\texttt{ivtrec}_\texttt{l}\ f\ a\ b\ c\ n + \texttt{ivtrec}_\texttt{r}\ f\ a\ b\ c\ n}{2}) > c.$$

The first conclusion becomes

$$f(\frac{\texttt{ivtrec}_\texttt{l}\ \texttt{f}\ \texttt{a}\ \texttt{b}\ \texttt{c}\ \texttt{n} + \texttt{ivtrec}_\texttt{r}\ \texttt{f}\ \texttt{a}\ \texttt{b}\ \texttt{c}\ \texttt{n}}{2}^{\uparrow}) > c \vee f(\frac{\texttt{ivtrec}_\texttt{l}\ \texttt{f}\ \texttt{a}\ \texttt{b}\ \texttt{c}\ \texttt{n} + \texttt{ivtrec}_\texttt{r}\ \texttt{f}\ \texttt{a}\ \texttt{b}\ \texttt{c}\ \texttt{n}}{2}^{\uparrow}) = c$$

which is proved as one of the disjuncts corresponds to the hypothesis added by the case split. The second conclusion becomes

$$f(\texttt{ivtrec}_\texttt{l}\ \texttt{f}\ \texttt{a}\ \texttt{b}\ \texttt{c}\ \texttt{n}) < c \vee f(\texttt{ivtrec}_\texttt{l}\ \texttt{f}\ \texttt{a}\ \texttt{b}\ \texttt{c}\ \texttt{n}) = c$$

which is the induction hypothesis, and hence strong fertilisation applies.

B.2 Inductive lemmas for Rolle's Theorem

- **Lemma (6.9)**

We yield proof-plan for the he inductive lemma

$$\forall n \in \mathbb{N}.\ \ \texttt{rolrec}_\texttt{r}\ f\ f'\ a\ b\ n - \texttt{rolrec}_\texttt{l}\ f\ f'\ a\ b\ n = \frac{b-a}{2^n}.$$

We employ the plan-specification for induction to produce a proof-plan for its lemma. We choose a standard structural induction scheme for the natural numbers, and choose single-step induction on *n*– the only candidate for induction.

The base case can be trivially proved by rewriting and identity, using the wave-rules defined for the partitioning function, the definition for exponentiation given by axiom (4.6) and the field equations for multiplication (4.15), (4.24) and (4.16).

$$\texttt{rolrec}_\texttt{r}\ f\ f'\ a\ b\ 0 - \texttt{rolrec}_\texttt{l}\ f\ f'\ a\ b\ 0 = \frac{b-a}{2^0}$$

can be rewritten to

$$b-a = \frac{b-a}{2^0}$$

which simplifies to

$$b-a = b-a$$

completing the proof-plan for the base case. The step case becomes

$$\texttt{rolrec}_\texttt{r}\ f\ f'\ a\ b\ \boxed{s(n)}^{\uparrow} - \texttt{rolrec}_\texttt{l}\ f\ f'\ a\ b\ \boxed{s(n)}^{\uparrow} = \frac{b-a}{2^{\boxed{s(n)}^{\uparrow}}}.$$

At this point we introduce four cases corresponding to the different conditions attached to the wave rules in figure 6.9. As an example, we describe here the proof-plan constructed for the first case in figure 6.9. After the application of arithmetical rules (4.9) and (4.15), the conclusion becomes

$$\boxed{\frac{\texttt{rolrec}_\texttt{l}\ f\ f'\ a\ b\ n + \boxed{\texttt{rolrec}_\texttt{r}\ f\ f'\ a\ b\ n}}{2}}^{\uparrow} - \texttt{rolrec}_\texttt{l}\ f\ f'\ a\ b\ n = \boxed{\frac{1}{2} \times \boxed{\frac{b-a}{2^n}}}^{\uparrow}.$$

Now we speculate the rule

$$\boxed{\frac{Y + \boxed{X}}{2}}^{\uparrow} - Y \quad \Rightarrow \quad \boxed{\frac{1}{2} \times \boxed{X - Y}}^{\uparrow} \tag{B.14}$$

which can now be used to rewrite the conclusion to

$$\boxed{\frac{1}{2} \times \boxed{\texttt{rolrec}_\texttt{r}\ f\ f'\ a\ b\ n - \texttt{rolrec}_\texttt{l}\ f\ f'\ a\ b\ n}}^{\uparrow} = \boxed{\frac{1}{2} \times \boxed{\frac{b-a}{2^n}}}^{\uparrow}$$

Now the proof-plan is completed by rewriting with the induction hypothesis, using the weak fertilisation method as described in section 3.2.5. At this point in the proof, the natural step is to cancel the $\frac{1}{2}$ terms on either side of the equality. In proof-planning terminology, this would allow strong fertilisation to take place- i.e. a direct invocation of the hypothesis. In our case we use weak fertilisation with the substitution rule of our logic, and then complete the proof using the reflexivity of equality.

- **Lemmas (6.10) and (6.11)**

We state these lemmas as

$$\forall n \in \mathbb{N}.\ \texttt{rolrec}_\texttt{r}\ f\ f'\ a\ b\ n \leq b \wedge \texttt{rolrec}_\texttt{r}\ f\ f'\ a\ b\ n \geq a$$

and

$$\forall n \in \mathbb{N}.\ \texttt{rolrec}_\texttt{l}\ f\ f'\ a\ b\ n \leq b \wedge \texttt{rolrec}_\texttt{l}\ f\ f'\ a\ b\ n \geq a$$

In order to yield proof-plans for these goals, we must use results from the proof of one conjecture in order to prove the other. In order to do this in λ*Clam*, we simulate a mutual induction scheme. This means that we perform induction on the conjunction of all the

goals, and then we can use the induction hypothesis of any of the conjuncts in order to rewrite the conclusion.

We state the conjecture for the left boundedness of the leftmost point of the partition

$$\forall n \in \mathbb{N}.\ \texttt{rolrec}_1\ f\ f'\ a\ b\ n > a \vee \texttt{rolrec}_1\ f\ f'\ a\ b\ n = a,$$

for which the base case is

$$\texttt{rolrec}_1\ f\ f'\ a\ b\ 0 > a \vee \texttt{rolrec}_1\ f\ f'\ a\ b\ 0 = a.$$

This is rewritten immediately to

$$a > a \vee a = a,$$

which is proved using the tautology method. The right boundedness condition is written

$$\forall n \in \mathbb{N}.\ \texttt{rolrec}_1\ f\ f'\ a\ b\ n < b \vee \texttt{rolrec}_1\ f\ f'\ a\ b\ n = b$$

for which the base case can be rewritten to

$$a < b \vee a = b,$$

and proved similarly.

The proof-plan for the base case of the left and right boundedness of the rightmost point follows the same steps so we omit the presentation of their proof-plans here.

The step case conclusion for the left boundedness of the left point of the partition is written

$$\texttt{rolrec}_1\ f\ f'\ a\ b\ s(n)^{\uparrow} > a \vee \texttt{rolrec}_1\ f\ f'\ a\ b\ s(n)^{\uparrow} = a,$$

and for the right point

$$\texttt{rolrec}_r\ f\ f'\ a\ b\ s(n)^{\uparrow} > a \vee \texttt{rolrec}_r\ f\ f'\ a\ b\ s(n)^{\uparrow} = a.$$

Now for the right boundedness for the left point the step case conclusion is written

$$\texttt{rolrec}_1\ f\ f'\ a\ b\ s(n)^{\uparrow} < b \vee \texttt{rolrec}_1\ f\ f'\ a\ b\ s(n)^{\uparrow} = b,$$

and for the right point

$$\texttt{rolrec}_r\ f\ f'\ a\ b\ s(n)^{\uparrow} < b \vee \texttt{rolrec}_r\ f\ f'\ a\ b\ s(n)^{\uparrow} = b.$$

Now we construct cases according to the conditions attached to the recursive wave rules for $\texttt{rolrec}_l$ and $\texttt{rolrec}_r$. Once again we consider the first case shown in figure 6.9. Using the conditions to the wave rules as cases is an example of **case analysis**. For this, we can rewrite two of these four conclusions immediately to

$$\texttt{rolrec}_l\ f\ f'\ a\ b\ n > a \vee \texttt{rolrec}_l\ f\ f'\ a\ b\ n = a$$
$$\texttt{rolrec}_l\ f\ f'\ a\ b\ n < b \vee \texttt{rolrec}_l\ f\ f'\ a\ b\ n = b$$

which can be solved as they correspond to the induction hypotheses.

Let us consider the step case conjunct for the rightboundedness of the right point:

$$\texttt{rolrec}_r\ f\ f'\ a\ b\ \boxed{s(n)}^{\uparrow} < b \vee \texttt{rolrec}_r\ f\ f'\ a\ b\ \boxed{s(n)}^{\uparrow} = b.$$

When the wave rules for this case are applied, we can embed the hypotheses for $\texttt{rolrec}_l$ and for $\texttt{rolrec}_r$. The annotated conclusion becomes

$$\boxed{\frac{\texttt{rolrec}_l\ f\ f'\ a\ b\ n + \texttt{rolrec}_r\ f\ f'\ a\ b\ n}{2}}^{\uparrow} < b \vee \boxed{\frac{\texttt{rolrec}_l\ f\ f'\ a\ b\ n + \texttt{rolrec}_r\ f\ f'\ a\ b\ n}{2}}^{\uparrow} = b.$$

We introduce the lemmas

$$\boxed{\frac{X+Y}{2}}^{\uparrow} > Z \vee \boxed{\frac{X+Y}{2}}^{\uparrow} = Z \Rightarrow \boxed{X > Z \vee X = Z} \wedge \boxed{Y > Z \vee Y = Z}^{\uparrow}$$
$$\boxed{\frac{X+Y}{2}}^{\uparrow} < Z \vee \boxed{\frac{X+Y}{2}}^{\uparrow} = Z \Rightarrow \boxed{X < Z \vee X = Z} \wedge \boxed{Y < Z \vee Y = Z}^{\uparrow},$$

which all pertain to the reasoning pattern we refer to as **division by 2**. We can rewrite the conclusion to

$$\boxed{\texttt{rolrec}_l\ f\ f'\ a\ b\ n < b \vee \texttt{rolrec}_l\ f\ f'\ a\ b\ n = b} \wedge \boxed{\texttt{rolrec}_r\ f\ f'\ a\ b\ n < b \vee \texttt{rolrec}_r\ f\ f'\ a\ b\ n = b}^{\uparrow}.$$

We can perform the same proof steps for the left boundedness conjecture and arrive at the conclusion

$$\boxed{\texttt{rolrec}_l\ f\ f'\ a\ b\ n > a \vee \texttt{rolrec}_l\ f\ f'\ a\ b\ n = a} \wedge \boxed{\texttt{rolrec}_r\ f\ f'\ a\ b\ n > a \vee \texttt{rolrec}_r\ f\ f'\ a\ b\ n = a}^{\uparrow}.$$

Now the subgoals are all proved for this branch of the case split, as strong fertilisation can now apply. The proof-plans for the other cases proceed similarly.

In the proof-plan for this lemma we used **mutual induction** techniques to introduce more than one induction hypothesis which allowed us to use **coloured rippling**.

- **Lemma (6.12)**

The inductive lemma we need to prove in order to establish this result is

$$\forall n \in \mathbb{N}.\ \mathtt{rolrec_r}\ f\ f'\ a\ b\ n > \mathtt{rolrec_l}\ f\ f'\ a\ b\ n.$$

We state the base case for this proof as

$$\mathtt{rolrec_r}\ f\ f'\ a\ b\ 0 > \mathtt{rolrec_l}\ f\ f'\ a\ b\ 0$$

which reduces to $b > a$, which follows trivially as it is in the hypotheses. The step case is written as

$$\mathtt{rolrec_r}\ f\ f'\ a\ b\ \boxed{s(n)}^{\uparrow} > \mathtt{rolrec_l}\ f\ f'\ a\ b\ \boxed{s(n)}^{\uparrow}$$

This can be proved independently of the partitioning criterion. In one set of cases the conclusion ripples to

$$\boxed{\frac{\mathtt{rolrec_l}\ f\ f'\ a\ b\ n + \mathtt{rolrec_r}\ f\ f'\ a\ b\ n}{2}}^{\uparrow} > \mathtt{rolrec_l}\ f\ f'\ a\ b\ n.$$

We use the rewrite rule

$$\frac{X+Y}{2} > Z \Rightarrow X > Z \wedge Y = Z \vee X = Z \wedge Y > Z \vee X > Z \wedge Y > Z$$

which pertains the reasoning pattern we refer to as **division by 2**. We rewrite the conclusion to

$$\begin{array}{l}
\mathtt{rolrec_l}\ f\ f'\ a\ b\ n > \mathtt{rolrec_l}\ f\ f'\ a\ b\ n \wedge \mathtt{rolrec_r}\ f\ f'\ a\ b\ n = \mathtt{rolrec_l}\ f\ f'\ a\ b\ n \vee \\
\mathtt{rolrec_l}\ f\ f'\ a\ b\ n = \mathtt{rolrec_l}\ f\ f'\ a\ b\ n \wedge \mathtt{rolrec_r}\ f\ f'\ a\ b\ n > \mathtt{rolrec_l}\ f\ f'\ a\ b\ n \vee \\
\mathtt{rolrec_l}\ f\ f'\ a\ b\ n > \mathtt{rolrec_l}\ f\ f'\ a\ b\ n \wedge \mathtt{rolrec_r}\ f\ f'\ a\ b\ n > \mathtt{rolrec_l}\ f\ f'\ a\ b\ n.
\end{array}$$

The second disjunct is satisfied by the induction hypothesis and the reflexivity of equality. In the other cases the proof-plans proceed similarly.

- **Lemma (6.13)**

We yield a proof-plan for the conjecture

$$\begin{array}{lll}
\forall n \in \mathbb{N}. & & \\
f'(\mathtt{rolrec_l}\ f\ f'\ a\ b\ n) \geq n & \wedge & f'(\mathtt{rolrec_r}\ f\ f'\ a\ b\ n) < n \quad \vee \\
f'(\mathtt{rolrec_l}\ f\ f'\ a\ b\ n) < n & \wedge & f'(\mathtt{rolrec_r}\ f\ f'\ a\ b\ n) \geq n \quad \vee \\
f'(\mathtt{rolrec_l}\ f\ f'\ a\ b\ n) \geq n & \wedge & f'(\mathtt{rolrec_r}\ f\ f'\ a\ b\ n) \geq n \\
 & \wedge & f(\mathtt{rolrec_l}\ f\ f'\ a\ b\ n) \geq f(\mathtt{rolrec_r}\ f\ f'\ a\ b\ n) \quad \vee \\
f'(\mathtt{rolrec_l}\ f\ f'\ a\ b\ n) < n & \wedge & f'(\mathtt{rolrec_r}\ f\ f'\ a\ b\ n) < n \\
 & \wedge & f(\mathtt{rolrec_l}\ f\ f'\ a\ b\ n) \leq f(\mathtt{rolrec_r}\ f\ f'\ a\ b\ n).
\end{array}$$

This lemma is the crucial lemma in establishing Rolle's Theorem. We consider the base case here first. This can immediately be rewritten to

$$(f'(a) < 0 \wedge f'(b) \geq 0) \ \vee \ (f'(a) \geq 0 \wedge f'(b) \geq 0 \wedge f(a) \geq f(b)) \ \vee$$
$$(f'(a) \geq 0 \wedge f'(b) < 0) \ \vee \ (f'(a) \leq 0 \wedge f'(b) \leq 0 \wedge f(a) \leq f(b)).$$

Now with a case analysis for the base case, we add $f'(a) < 0 \vee f'(a) \geq 0$ and $f'(b) < 0 \vee f'(b) \geq 0$ which splits into four cases when $\vee_l$ is applied twice. Then since $f(a) = f(b)$ is in the hypotheses, each of the conjuncts on one of the disjuncts is true in every case.

For the step case, we apply the case-split set for Rolle's Theorem. In one case we add the following to the hypotheses:

$$f((\mathtt{rolrec_l}\ f\ f'\ a\ b\ n + \mathtt{rolrec_r}\ f\ f'\ a\ b\ n)/2) \geq f(\mathtt{rolrec_l}\ f\ f'\ a\ b\ n) \ \wedge$$
$$f'((\mathtt{rolrec_l}\ f\ f'\ a\ b\ n + \mathtt{rolrec_r}\ f\ f'\ a\ b\ n)/2) \geq 0.$$

Now we need to satisfy one disjunct in the lemma (6.13). In this case the disjunct of the step case which is satisfied is

$$f'(\mathtt{rolrec_l}\ f\ f'\ a\ b\ s(n)^{\uparrow}) \geq 0 \ \wedge$$
$$f'(\mathtt{rolrec_r}\ f\ f'\ a\ b\ s(n)^{\uparrow}) < 0,$$

whence the step case then becomes

$$f'(\frac{\mathtt{rolrec_l}\ f\ f'\ a\ b\ n + \mathtt{rolrec_r}\ f\ f'\ a\ b\ n}{2}^{\uparrow}) \geq 0 \ \wedge$$
$$f'(\mathtt{rolrec_r}\ f\ f'\ a\ b\ n) < 0.$$

Now one conjunct matches one of the conditions, and the other conjunct matches a hypothesis.

Now we consider a different case from the case-split set, where we add the following term to the hypotheses:

$$f((\mathtt{rolrec_l}\ f\ f'\ a\ b\ n + \mathtt{rolrec_r}\ f\ f'\ a\ b\ n)/2) \geq f(\mathtt{rolrec_l}\ f\ f'\ a\ b\ n) \ \wedge$$
$$f'((\mathtt{rolrec_l}\ f\ f'\ a\ b\ n + \mathtt{rolrec_r}\ f\ f'\ a\ b\ n)/2) \geq 0.$$

In this case the disjunct of the step case which is satisfied is

$$f'(\mathtt{rolrec_l}\ f\ f'\ a\ b\ s(n)^{\uparrow}) \geq 0 \ \wedge$$
$$f'(\mathtt{rolrec_r}\ f\ f'\ a\ b\ s(n)^{\uparrow}) \geq 0 \ \wedge$$
$$f(\mathtt{rolrec_l}\ f\ f'\ a\ b\ s(n)^{\uparrow}) \geq f(\mathtt{rolrec_r}\ f\ f'\ a\ b\ s(n)^{\uparrow})$$

which under this condition rewrites to

$$f'(\uparrow\frac{\texttt{rolrec}_\texttt{l}\ f\ f'\ a\ b\ n + \texttt{rolrec}_\texttt{r}\ f\ f'\ a\ b\ n2}{2}) \geq 0\ \wedge$$
$$f'(\texttt{rolrec}_\texttt{r}\ f\ f'\ a\ b\ n) \geq 0\ \wedge$$
$$f(\uparrow\frac{\texttt{rolrec}_\texttt{l}\ f\ f'\ a\ b\ n + \texttt{rolrec}_\texttt{r}\ f\ f'\ a\ b\ n}{2}) \geq f(\texttt{rolrec}_\texttt{r}\ f\ f'\ a\ b\ n).$$

Now we appeal to the fact that each of the conjuncts is either a condition or a hypothesis. The proof-plans all proceed similarly for each of the other cases.

B.3 Intermediate lemmas

Here we state a number of lemmas which were used in the proofs, and which we incorporated in to λ*Clam* interactively. As the proofs of the inductive lemmas for Rolle's Theorem and the Intermediate Value Theorem involve many case splits, each branch needs only slightly different lemmas, which we group into families. We introduce these lemmas interactively into the system as rewrite rules used often in the proofs. The rules which we use to rewrite the hypotheses are presented with a logical implication $\rightarrow$, as this is the direction of rewriting on the hypotheses. Those that we use to rewrite the goal, we present using the rewriting symbol $\Rightarrow$. The rules appear to be duplicated, as we have incorporated symmetry. We have

$$A > 0 \wedge B > 0 \quad \rightarrow \quad \frac{A}{B} > 0 \tag{B.15}$$
$$A > 0 \wedge B < 0 \quad \rightarrow \quad \frac{A}{B} < 0 \tag{B.16}$$
$$A < 0 \wedge B > 0 \quad \rightarrow \quad \frac{A}{B} < 0 \tag{B.17}$$
$$A < 0 \wedge B < 0 \quad \rightarrow \quad \frac{A}{B} > 0 \tag{B.18}$$
$$A \approx B \wedge A < C \wedge B > C \quad \rightarrow \quad A \approx C \tag{B.19}$$
$$A \approx B \wedge A < C \wedge B > C \quad \rightarrow \quad B \approx C \tag{B.20}$$

which help when ascertaining the sign of derivatives. We also add the following rules

$$A \approx B \wedge A > X \rightarrow B \approx X \vee B > X$$
$$B \approx A \wedge A > X \rightarrow B \approx X \vee B > X$$
$$A \approx B \wedge A < X \rightarrow B \approx X \vee B < X$$
$$B \approx A \wedge A < X \rightarrow B \approx X \vee B < X$$

which allow us to determine the infinitesimal neighbourhood of points for which we have order constraints. We also introduce the following lemmas as wave rules

$$\boxed{\frac{X+Y}{2}}^{\uparrow} > Z \vee \boxed{\frac{X+Y}{2}}^{\uparrow} = Z \Rightarrow \boxed{X > Z \vee X = Z} \wedge \boxed{Y > Z \vee Y = Z}^{\uparrow}$$

$$\boxed{\frac{X+Y}{2}}^{\uparrow} < Z \vee \boxed{\frac{X+Y}{2}}^{\uparrow} = Z \Rightarrow \boxed{X < Z \vee X = Z} \wedge \boxed{Y < Z \vee Y = Z}^{\uparrow}.$$

We add another set of rewrite rules

$$\frac{X+Y}{2} > Z \Rightarrow X > Z \wedge Y = Z \vee X = Z \wedge Y > Z \vee X > Z \wedge Y > Z$$

$$\frac{X+Y}{2} < Z \Rightarrow X < Z \wedge Y = Z \vee X = Z \wedge Y < Z \vee X < Z \wedge Y < Z.$$

Bibliography

[Abian, 1979] Abian, A. (1979). An ultimate proof of rolle's theorem. *American Mathematical Monthly*, 86(6):484–485.

[Adams et al., 1999] Adams, A. A., Gottliebsen, H., Linton, S. A., and Martin, U. (1999). VSDITLU: a verifiable symbolic definite integral table look-up. In *Conference on Automated Deduction*, pages 112–126.

[Apostol, 1974] Apostol, T. (1974). *Mathematical Analysis: A Modern Approach to Advanced Calculus (2nd Edition)*. Addison-Wesley Publishing Company.

[Aubin, 1976] Aubin, R. (1976). Mechanizing structural induction. PhD thesis, University of Edinburgh.

[Aubin, 1979] Aubin, R. (1979). Mechanizing structural induction. part I: Formal system. part II: Strategies. *Theoretical Computer Science*, 1(9):329–362.

[Baader and Nipkow, 1998] Baader, F. and Nipkow, T. (1998). *Term Rewriting and All That*. Cambridge University Press.

[Basin and Walsh, 1996] Basin, D. and Walsh, T. (1996). A calculus for and termination of rippling. *Journal of Automated Reasoning*, 16(1–2):147–180.

[Bedrax, 1993] Bedrax, T. (1993). **Infmal**: Prototype of an interactive theorem prover based on infinitesimal analysis. Master's thesis, Pontificia Universidad Católica de Chile.

[Beeson, 1985] Beeson, M. (1985). *Foundations of Constructive Mathematics*. Springer, Berlin/Heidelberg/New York. Metamathematical Studies.

[Beeson, 1995] Beeson, M. (1995). Using nonstandard analysis to verify the correctness of computations. *International Journal of Foundations of Computer Science*, 6(3):299–338.

[Beeson, 1998] Beeson, M. (1998). Automatic generation of epsilon-delta proofs of continuity. In Calmet, J. and Plaza, J. A., editors, *AISC*, volume 1476 of *Lecture Notes in Computer Science*, pages 67–83. Springer.

[Bell, 1998] Bell, J. L. (1998). *A Primer of Infinitesimal Analysis*. Cambridge University Press.

[Benzmüller et al., 1997] Benzmüller, C., Cheikhrouhou, L., Fehrer, D., Fiedler, A., Huang, X., Kerber, M., Kohlhase, K., Meier, A., Melis, E., Schaarschmidt, W., Siekmann, J., and Sorge, V. (1997). Ωmega: Towards a mathematical assistant. In McCune, W., editor, *14th International Conference on Automated Deduction*, pages 252–255. Springer-Verlag.

[Berkeley, 1734] Berkeley, G. (1734). *The Analyst*: A discourse addressed to an infidel mathematician. Available from `http://www.maths.tcd.ie/pub/HistMath/People/Berkeley/Analyst/`.

[Bishop and Bridges, 1985] Bishop, E. and Bridges, D. (1985). *Constructive Analysis*. Springer-Verlag.

[Bledsoe, 1990] Bledsoe, W. W. (1990). Challenge problems in elementary calculus. *Journal of Automated Reasoning*, 6(3):341–359.

[Bledsoe and Ballantyne, 1977] Bledsoe, W. W. and Ballantyne, A. M. (1977). Automatic proofs of theorems in analysis using nonstandard techniques. *Association for Computing Machinery*, 24(3):353–374.

[Bledsoe et al., 1972] Bledsoe, W. W., Boyer, R. S., and Henneman, W. H. (1972). Computer proofs of limit theorems. *Artificial Intelligence*, 3:27–60.

[Boulton et al., 1998] Boulton, R., Slind, K., Bundy, A., and Gordon, M. (1998). An interface between CLAM and HOL. In Grundy, J. and Newey, M., editors, *Proceedings of the 11th International Conference on Theorem Proving in Higher Order Logics (TPHOLs'98)*, volume 1479 of *Lecture Notes in Computer Science*, pages 87–104, Canberra, Australia. Springer.

[Boyer and Moore, 1990] Boyer, R. S. and Moore, J. S. (1990). A theorem prover for a computational logic. In *Proceedings of the Tenth International Conference on Automated Deduction*. Kaiserlauten, Germany.

[Bundy, 1988] Bundy, A. (1988). The use of explicit plans to guide inductive proofs. In Lusk, R. and Overbeek, R., editors, *9th International Conference on Automated Deduction*, pages 111–120. Springer-Verlag. Longer version available from Edinburgh as DAI Research Paper No. 349.

[Bundy, 1989] Bundy, A. (1989). A science of reasoning. Research Paper 445, Dept. of Artificial Intelligence, University of Edinburgh. Also in "Computational Logic: Essays in Honor of Alan Robinson", MIT Press, 1991.

[Bundy et al., 1993] Bundy, A., Stevens, A., van Harmelen, F., Ireland, A., and Smaill, A. (1993). Rippling: A heuristic for guiding inductive proofs. *Artificial Intelligence*, 62:185–253. Also available from Edinburgh as DAI Research Paper No. 567.

[Bundy et al., 1990] Bundy, A., van Harmelen, F., Horn, C., and Smaill, A. (1990). The Oyster-Clam system. Research Paper 507, Dept. of Artificial Intelligence, University of Edinburgh. Appeared in the proceedings of CADE-10.

[Cantu et al., 1996] Cantu, F., Bundy, A., Smaill, A., and Basin, D. (1996). Experiments in automating hardware verification using inductive proof planning. In Srivas, M. and Camilleri, A., editors, *Proceedings of the Formal Methods for Computer-Aided Design Conference*, number 1166 in Lecture Notes in Computer Science, pages 94–108. Springer-Verlag.

[Castellini and Smaill, 2001] Castellini, C. and Smaill, A. (2001). Tactic-based theorem proving in first-order modal and temporal logics. In Giunchiglia, E. and Massacci, F., editors, *Proceedings of IJCAR WS10: Issues in the Design and Experimental Evaluation of Systems for Modal and Temporal Logics*, TR DII 14/01. University of Siena.

[Castellini and Smaill, 2002] Castellini, C. and Smaill, A. (2002). Proof planning for feature interactions: A preliminary report. In Baaz, M. and Voronkov, A., editors, *Proceedings of LPAR 2002*, LNAI 2514, pages 102–114. Springer.

[Chippendale, 1995] Chippendale, M. (1995). Planning a proof of the intermediate value theorem. In Calmet, J. and Campbell, J. A., editors, *Integrating Symbolic Mathematical Computation and Artificial Intelligence, Second International Conference, AISMC-2*, volume 958 of *Lecture Notes in Computer Science*. Springer.

[Church, 1940] Church, A. (1940). A formulation of the simple theory of types. *Symbolic Logic*, 5(1):56–68.

[Cruz-Filipe, 2002] Cruz-Filipe, L. (2002). Formalizing real calculus in coq. In Carreño, V., Muñoz, C., and Tahar, S., editors, *Theorem Proving in Higher Order Logics*, NASA Conference Proceedings, Hampton VA.

[Dixon and Fleuriot, 2003] Dixon, L. and Fleuriot, J. D. (2003). IsaPlanner: A prototype proof planner in Isabelle. In *Proceedings of CADE'03*, Lecture Notes in Computer Science.

[Dutertre, 1996] Dutertre, B. (1996). Elements of mathematical analysis in pvs. In von Wright, J., Grundy, J., and Harrison, J., editors, *TPHOLs*, volume 1125 of *Lecture Notes in Computer Science*, pages 141–156. Springer.

[Fleuriot, 2001a] Fleuriot, J. (2001a). *A Combination of Geometry Theorem Proving and Nonstandard Analysis, with Application to Newton's Principia.* Springer-Verlag.

[Fleuriot and Paulson, 2000] Fleuriot, J. and Paulson, L. (2000). Mechanizing Nonstandard Real Analysis. To appear in LMS Journal of Computation and Mathematics.

[Fleuriot, 2000] Fleuriot, J. D. (2000). On the mechanization of real analysis in Isabelle/HOL. In Harrison, J. and Aagaard, M., editors, *Theorem Proving in Higher Order Logics: 13th International Conference, TPHOLs 2000*, volume 1869 of *Lecture Notes in Computer Science*, pages 146–162. Springer-Verlag.

[Fleuriot, 2001b] Fleuriot, J. D. (2001b). Nonstandard geometric proofs. In Wang, D. and Richter-Gebert, J., editors, *Automated Deduction in Geometry*, volume 2061, pages 238–260. A shorter version appeared in the published in Proceedings of ADG 2000, pages 259–279, Zurich, Switzerland.

[Fleuriot, 2001c] Fleuriot, J. D. (2001c). Theorem proving in infinitesimal geometry. *Logic Journal of the IGPL*, 9(3):471–498.

[Fleuriot and Paulson, 1999] Fleuriot, J. D. and Paulson, L. C. (1999). Proving Newton's Propositio Kepleriana using geometry and nonstandard analysis in Isabelle. In *Automated Deduction in Geometry 1998*, volume 1669 of *Lecture Notes in Artificial Intelligence*, pages 47–66.

[Franke et al., 1999] Franke, A., Hess, S. M., Jung, C. G., Kohlhase, M., and Sorge, V. (1999). Agent-oriented integration of distributed mathematical services. *Journal of Universal Computer Science*, 5:156–187.

[Franke and Kohlhase, 1999] Franke, A. and Kohlhase, M. (1999). System description: MATHWEB, an agent-based communication layer for distributed automated theorem proving. In Ganzinger, H., editor, *CADE*, volume 1632 of *Lecture Notes in Computer Science*, pages 217–221. Springer.

[Gamboa, 1999] Gamboa, R. (1999). *Mechanically Verifying Real-Valued Algorithms in ACL2*. PhD thesis, The University of Texas at Austin.

[Goldblatt, 1991] Goldblatt, R. (1991). *Lectures on the hyperreals*. Springer. Graduate Texts in Mathematics.

[Gordon and Melham, 1993] Gordon, M. J. C. and Melham, T. F., editors (1993). *Introduction to HOL: A theorem proving environment for higher order logic*. Cambridge University Press.

[Gottliebsen, 2000] Gottliebsen, H. (2000). Transcendental functions and continuity checking in PVS. In Aargaard, M. and Harrison, J., editors, *Theorem Proving in Higher Order Logics: 13th International Conference, TPHOLs 2000*, volume 1869, pages 197–214, Portland, OR. Springer-Verlag.

[Harrison, 1992] Harrison, J. (1992). Constructing the real numbers in HOL. In Claesen, L. J. M. and Gordon, M. J. C., editors, *Proceedings of the IFIP TC10/WG10.2 International Workshop on Higher Order Logic Theorem Proving and its Applications*, volume A-20 of *IFIP Transactions A: Computer Science and Technology*, pages 145–164, IMEC, Leuven, Belgium. North-Holland.

[Harrison, 1998] Harrison, J. (1998). *Theorem proving with the real numbers*. PhD thesis, University of Cambridge.

[Harrison, 1999] Harrison, J. (1999). A machine-checked theory of floating point arithmetic. In Bertot, Y., Dowek, G., Hirschowitz, A., Paulin, C., and Théry, L., editors, *Theorem Proving in Higher Order Logics: 12th International Conference, TPHOLs'99*, volume 1690 of *Lecture Notes in Computer Science*, pages 113–130, Nice, France. Springer-Verlag.

[Heneveld et al., 2001] Heneveld, A., Maclean, E., Bundy, A., Fleuriot, J., and Smaill, A. (2001). Solving integrals at the method level. In Kerber, M. and Kohlhase, M., editors, *Symbolic Computation and Automated Reasoning*, pages 251–252.

[Henzinger et al., 1997] Henzinger, T. A., Ho, P.-H., and Wong-Toi, H. (1997). HYTECH: A model checker for hybrid systems. *International Journal on Software Tools for Technology Transfer*, 1(1-2):110–122.

[Henzinger et al., 1992] Henzinger, T. A., Nicollin, X., Sifakis, J., and Yovine, S. (1992). Symbolic Model Checking for Real-Time Systems. In *7th. Symposium of Logics in Computer Science*, pages 394–406, Santa-Cruz, California. IEEE Computer Scienty Press.

[Hoskins, 1990] Hoskins, R. F. (1990). *Standard and Nonstandard Analysis*. Mathematics and its applications. Ellis Horwood Limited.

[Hurd and Loeb, 1985] Hurd, A. E. and Loeb, P. A. (1985). *An Introduction to Nonstandard Real Analysis*, volume 118 of *Pure and Applied Mathematics*. Academic Press Inc.

[Hutter, 1997] Hutter, D. (1997). Colouring terms to control equational reasoning. *Journal of Automated Reasoning*, 18:399–442.

[Hutter and Kohlhase, 1997] Hutter, D. and Kohlhase, M. (1997). A colored version of the λ-Calculus. In McCune, W., editor, *14th International Conference on Automated Deduction*, pages 291–305. Springer-Verlag. Also available as SEKI-Report SR-95-05.

[Ireland, 1992] Ireland, A. (1992). The Use of Planning Critics in Mechanizing Inductive Proofs. In Voronkov, A., editor, *International Conference on Logic Programming and Automated Reasoning – LPAR 92, St. Petersburg*, Lecture Notes in Artificial Intelligence No. 624, pages 178–189. Springer-Verlag. Also available from Edinburgh as DAI Research Paper 592.

[Ireland and Bundy, 1996] Ireland, A. and Bundy, A. (1996). Productive use of failure in inductive proof. *Journal of Automated Reasoning*, 16(1–2):79–111. Also available as DAI Research Paper No 716, Dept. of Artificial Intelligence, Edinburgh.

[Janičić and Bundy, 2002] Janičić, P. and Bundy, A. (2002). A general setting for the flexible combining and augmenting decision procedures. *Journal of Automated Reasoning*, 28(3):257–305.

[Janičić et al., 1999] Janičić, P., Bundy, A., and Green, I. (1999). A framework for the flexible integration of a class of decision procedures into theorem provers. In E., G., editor, *Proceedings of CADE-16*, pages 127–141, Trento, Italy. Springer-Verlag. LNAI 1652.

[Kaufmann and Moore, 1997] Kaufmann, M. and Moore, J. S. (1997). An industrial strength theorem prover for a logic based on common lisp. *IEEE Transactions on Software Engineering*, 23(4):203–213.

[Keisler, 1977] Keisler, H. J. (1977). *Foundations of Infinitesimal Calculus*. Prindle, Weber & Schmidt.

[Liu, 1980] Liu, S.-C. (1980). A proof-theoretic approach to non-standard analysis with emphasis on distinguishing between constructive and non-constructive results. *The Kleene Symposium*, pages 391–414.

[Lowe and Duncan, 1997] Lowe, H. and Duncan, D. (1997). XBarnacle: Making theorem provers more accessible. In McCune, W., editor, *14th International Conference on Automated Deduction*, pages 404–408. Springer-Verlag.

[Maclean, 2001] Maclean, E. (2001). Automating proof in non-standard analysis. In Striegnitz, K., editor, *Proceedings of the Student Session of the 13th European Summer School in Logic, Language and Information*, pages 191–203.

[Maclean, 2003] Maclean, E. (2003). Proof-planning non-standard analysis. *Submitted to the Annals of Artificial Intelligence and Mathematics (March 2002)*.

[Maclean et al., 2002] Maclean, E., Fleuriot, J., and Smaill, A. (2002). Proof-planning non-standard analysis. In *Proceedings of the Seventh International Symposium on Artificial Intelligence and Mathematics*, Fort Lauderdale, Florida.

[Mayero, 2001] Mayero, M. (2001). *Formalisation et automatisation de preuves en analyses relle et numérique*. PhD thesis, Université de Paris VI.

[Melis, 1996] Melis, E. (1996). Progress in proof planning: Planning limit theorems automatically. Technical Report SR-97-08, University of the Saarland.

[Melis, 1998] Melis, E. (1998). AI-Techniques in Proof Planning. In Henri Prade, editor, *13th European Conference on Artificial Intelligence*, pages 494–498, Brighton, UK. John Wiley and Sons, Chichester.

[Miller and Nadathur, 1988] Miller, D. and Nadathur, G. (1988). An overview of λProlog. In Bowen, K. & Kowalski, R., editor, *Proceedings of the Fifth International Logic Programming Conference/ Fifth Symposium on Logic Programming*. MIT Press.

[Nelson, 1977] Nelson, E. (1977). Internal set theory: A new approach to nonstandard analysis. *Bulletin American Mathematical Society*, 83. Available from `http://www.math.princeton.edu/~nelson/books.html`.

[Owre et al., 1992] Owre, S., Rushby, J. M., and Shankar, N. (1992). PVS : An integrated approach to specification and verification. Tech report, SRI International.

[Palmgren, 1995] Palmgren, E. (1995). A Constructive Approach to Nonstandard Analysis. *Annals of Pure and Applied Logic*, 73(3):297–325.

[Palmgren, 1997] Palmgren, E. (1997). A Sheaf-Theoretic Foundation for Nonstandard Analysis. *Annals of Pure and Applied Logic*, 85(1):69–86.

[Paulson, 1989] Paulson, L. (1989). The foundation of a generic theorem prover. *Journal of Automated Reasoning*, 5:363–397.

[Richardson et al., 1998] Richardson, J. D. C., Smaill, A., and Green, I. (1998). System description: proof planning in higher-order logic with Lambda-Clam. In Kirchner, C. and Kirchner, H., editors, *15th International Conference on Automated Deduction*, volume 1421 of *Lecture Notes in Artificial Intelligence*, pages 129–133, Lindau, Germany.

[Robinson, 1966] Robinson, A. (1966). *Non-standard Analysis*. North-Holland Publishing Company, Amsterdam. Studies in Logic and the Foundations of Mathematics.

[Robinson, 1996] Robinson, A. (1996). *Non-standard Analysis*. Princeton University Press, Princeton, New Jersey. Revised Edition.

[Simpson, 1990] Simpson, A. P. (1990). The infidel is innocent. *The Mathematical Intelligencer*, 12(3).

[Smaill and Green, 1996] Smaill, A. and Green, I. (1996). Higher-order annotated terms for proof search. In von Wright, J., Grundy, J., and Harrison, J., editors, *Theorem Proving in Higher Order Logics: 9th International Conference, TPHOLs'96*, volume 1275 of *Lecture Notes in Computer Science*, pages 399–414, Turku, Finland. Springer-Verlag. Also available as DAI Research Paper 799.

[Stark and Ireland, 1998] Stark, J. and Ireland, A. (1998). Invariant discovery via failed proof attempts. In Flener, P., editor, *Logic-based Program Synthesis and Transformation*, number 1559 in LNCS, pages 271–288. Springer-Verlag.

[Troelstra, 1973] Troelstra, A. S., editor (1973). *Metamathematical Investigation of Intuitionistic Arithmetic and Analysis*. Number 344 in Lecture Notes in Mathematics. Springer-Verlag.

[van Benthem Jutting, 1977] van Benthem Jutting, L. (1977). *Checking Landau's "Grundlagen" in the Automath system*. PhD thesis, Technische Hogeschool Eindhoven, Stichting Mathematisch Centrum.

[Walsh et al., 1992] Walsh, T., Nunes, A., and Bundy, A. (1992). The use of proof plans to sum series. In Kapur, D., editor, *11th International Conference on Automated Deduction*, pages 325–339. Springer Verlag. Lecture Notes in Computer Science No. 607. Also available from Edinburgh as DAI Research Paper 563.

[Yoshida, 1993] Yoshida, T. (1993). Coloured rippling. Master's thesis, Dept. of Artificial Intelligence, University of Edinburgh.

[Yoshida et al., 1994] Yoshida, T., Bundy, A., Green, I., Walsh, T., and Basin, D. (1994). Coloured rippling: An extension of a theorem proving heuristic. In Cohn, A. G., editor, *Proceedings of ECAI-94*, pages 85–89. John Wiley.